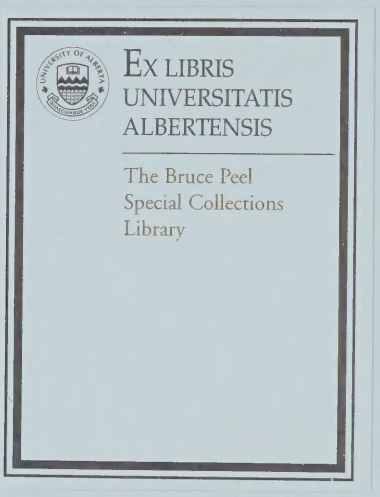
EX LIBRIS
UNIVERSITATIS
ALBERTENSIS
The Bruce Peel
Special Collections
Library

University of Alberta

Library Release Form

Name of Author : Dexter Odchigue Cahoy

Title of Thesis : Finite Element Methods for Bivariate Smoothing

Degree : Master of Science

Year this Degree Granted : 2003

Permission is hereby granted to the University of Alberta Library to reproduce single copies of this thesis and to lend or sell such copies for private, scholarly or scientific research purposes only.

The author reserves all other publication and other rights in association with the copyright in the thesis, and except as herein before provided, neither the thesis nor any substantial portion thereof may be printed or otherwise reproduced in any material form whatever without the author's prior written permission.

University of Alberta

Finite Element Methods for Bivariate Smoothing

by

Dexter Odchigue Cahoy

A thesis submitted to the Faculty of Graduate Studies and Research in partial fulfillment of the requirements for the degree of Master of Science

in

Statistics

Department of Mathematical and Statistical Sciences

Edmonton, Alberta

Fall 2003

Digitized by the Internet Archive
in 2025 with funding from
University of Alberta Library

https://archive.org/details/162017992619

UNIVERSITY OF ALBERTA

Faculty of Graduate Studies and Research

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies and Research for acceptance, a thesis entitled **Finite Element Methods for Bivariate Smoothing** submitted by **Dexter Odchigue Cahoy** in partial fulfillment of the requirements for the degree of Master of Science in Statistics.

Abstract

Finite element method (FEM) is used for approximating solutions in bivariate smoothing. Finite Element (FE) approximations are implemented with linear functions as the interpolants. These approximations are formulated based on the thin-plate spline and L-spline smoothing problems. A modified FE L-spline (MFE L-spline) smoothing method is also proposed and compared with the finite element L-spline (FE L-spline) and finite element thin-plate spline (TPSFEM) methods. These FE-based methods are compared with the standard thin-plate spline as a benchmark. The corresponding algorithms are implemented in Matlab and documented by Noweb. Generalized FE procedures based on penalized weighted residual sum of squares are also proposed and implemented for bivariate smoothing. Possible localization properties of these methods, which are helpful in capturing local features of the data are discussed. FE-based methods are used to smooth actual and real data sets with interpretation of the resulting smooths. Smooths constructed using generalized cross-validation (GCV) method are also presented.

Acknowledgment

I am very grateful to my supervisor, *Dr. Ivan Mizera*, for providing me the necessary support for this project, for his untiring help, invaluable guidance and very helpful criticisms.

I am also indebted to my panelists *Dr. Hee-Seok Oh* and *Dr. Karen Tomic* for helping me improve my project; *Ms. Dona Guelzow* for her unselfish assistance; *Dr. Doug Wiens*, *Dr. Subhash Lele* and *Dr. K.C. Carriere* for their inspiring advice and invaluable support. I am also thankful to all my professors who have heightened my knowledge in statistics.

A special thanks also goes to *Dr. Akbar Rhemtulla* for the helpful tips on how to live in Edmonton and to the Department of Mathematical and Statistical Sciences especially *Dr. Jim Muldowney* and *Sir Rick Mikalonis* for the Teaching Assistantships.

To my parents, brother and sisters, for their support and inspiration; to my parents-in-law, for helping me achieve my goals and ambitions and for taking care of my beloved daughter; and to *Jay* and *Karen Barcelon* for the financial support.

To the *KNOX-EFC* Church for their moral support; to *Zorik, Boyan, Mark* and *Yulia* for sharing their expertise in FEM and LaTeX; and to all my Filipino friends.

To *Jim, Louise* and *Steven Peck* for their encouragement, comfort and help.

To my wife *Armi* and daughter *Yssa*, for all their endless support, full cooperation and love.

Most of all, to our Holy Father, our GOD, for all the blessings, courage and wisdom.

Dedication

To my beloved family:

Alyssa Bernadette Villarente Cahoy (*Diday*, Daughter)

and

Armi Viacrucis Villarente Cahoy(*Mik*, Wife)

whose inspiration, advice, care and love are greatly valued.

Table of Contents

1 Introduction 1

2 Principles of Smoothing via Roughness Penalty Approach 6

2.1 (Parametric) Regression Approaches 6

2.1.1 (Simple) Linear Regression 6

2.1.2 Polynomial Regression 7

2.2 Spline Smoothing 7

2.2.1 Spline Functions in Smoothing 7

2.2.2 Univariate Spline Smoothing 9

2.2.3 Minimizer of the Penalized Least Squares 9

2.3 Extensions of Penalized Least Squares 11

2.3.1 From Univariate to Bivariate Smoothing 11

2.3.2 Squared Laplacian Penalty for L-spline Surface Fitting 13

3 Smooth Approximation by Finite Element Method (FEM) 15

3.1 Function Spaces 15

3.1.1 Certain Hilbert Sobolev Spaces 15

3.1.2 Usual Inner Products in $\mathcal{H}^m(\Omega)$ Spaces 16

3.2 FEM via Galerkin Formulation 17
3.2.1 The Basis 20
3.3 Finite Element(FE) Formulation of Bivariate Smoothing Problems . . 22
3.3.1 FE L-spline Formulation 22
3.3.2 FE Thin-Plate Spline (TPSFEM) Formulation 23
3.4 Setting Up the Linear System 24
3.4.1 Bivariable FE L-spline Set-Up 24
3.4.2 Bivariable TPSFEM Set-Up 25
3.5 Modified FE Formulation of the Bivariate L-spline Smoothing Problem 28
3.5.1 Modified FE L-spline (MFE L-spline) Formulation 28
3.5.2 Setting Up the Bivariable MFE L-spline's Linear System . . . 30
3.6 FE Formulation of Generalized Bivariate Smoothing Problems 31
3.6.1 The Construction of a Generalized Smooth : A Sample 31

4 Implementation, Testing and Interpretation of Results 35

4.1 The Assembly 35
4.2 Applications 38
4.2.1 Synthetic Data: Wendelberger Function 38
4.2.2 Generalized Smooths for Synthetic Data: Wendelberger Function 45
4.2.3 Real Data Sets 49
4.2.4 Generalized Smooths for Ozone Data 49
4.2.5 Generalized Smooths for Cobar Mine Data 55
4.3 Interpretation of Smooths from Real Data Sets 64
4.3.1 Ozone Data 64
4.3.2 Cobar Mine Data 67
4.4 Parameter (λ) Selection by Generalized Cross-Validation (GCV) Method 69

5 Summary, Conclusions and Future Research Directions **72**

5.1 Summary 72

5.2 Conclusions 74

5.3 Possible Research Directions 74

Bibliography **76**

Appendix : NOWEB-Processed Documentation **80**

List of Figures

3.1 Linear Shape Function N_j for (local) Node 1 of Element η_j. 21

3.2 Basis Function ψ_j for a Cluster of Five Elements at Global Node j. . 22

4.1 Graph of the Wendelberger Function. 39

4.2 FE L-spline's Bivariate Fits of the Wendelberger Function as $\lambda \to \infty$. 41

4.3 MFE L-spline's Bivariate Fits of the Wendelberger Function as $\lambda \to \infty$. 42

4.4 TPSFEM's Bivariate Fits of the Wendelberger Function as $\lambda \to \infty$. . 43

4.5 Thin-Plate Spline Fits of the Wendelberger Function as $\lambda \to \infty$. . . . 44

4.6 FE L-Spline's Bivariate Smooths of the Noisy Wendelberger Function. 45

4.7 MFE L-Spline's Bivariate Smooths of the Noisy Wendelberger Function. 46

4.8 TPSFEM's Bivariate Smooths of the Noisy Wendelberger Function. . 47

4.9 Thin-Plate Spline's Smooths of the Noisy Wendelberger Function. . . 48

4.10 FE L-spline's Contour Plots of Ozone Data as $\lambda \to \infty$. 51

4.11 MFE L-spline's Contour Plots of Ozone Data as $\lambda \to \infty$. 52

4.12 TPSFEM's Contour Plots of Ozone Data as $\lambda \to \infty$. 53

4.13 Thin-Plate Spline's Contour Plots of Ozone Data as $\lambda \to \infty$. 54

4.14 FE L-spline's Bivariate Smooths of the Cobar Mine Data as $\lambda \to \infty$. 56

4.15 MFE L-spline's Bivariate Smooths of the Cobar Mine Data as $\lambda \to \infty$. 57

4.16 TPSFEM's Bivariate Smooths of the Cobar Mine Data as $\lambda \to \infty$. . . 58
4.17 Thin-Plate Spline's Bivariate Fits of the Cobar Mine Data as $\lambda \to \infty$. 59
4.18 FE L-spline's Contour Plots of the Cobar Mine Data as $\lambda \to \infty$. . . . 60
4.19 MFE L-spline's Contour Plots of the Cobar Mine Data as $\lambda \to \infty$. . . 61
4.20 TPSFEM's Contour Plots of the Cobar Mine Data as $\lambda \to \infty$. 62
4.21 Thin-Plate Spline's Contour Plots of the Cobar Mine Data as $\lambda \to \infty$. 63
4.22 Contour Plots of Ozone Data where λ is Computed by GCV method. 70
4.23 Contours of Cobar Mine Data where λ is Computed by GCV method. 71

Chapter 1

Introduction

Finite element method (FEM) has been used for several decades in various scientific fields, particularly in solving complex engineering problems such as partial differential equations (PDE). It has also been known to provide an efficient numerical procedure for solving scientific problems. In statistics, its use dates only very recently; so far, it has been mostly used for approximating solutions of nonparametric regression problems.

In nonparametric regression, we model data as

$$observed\ \ value = true\ \ value\ +\ \ error, \tag{1.1}$$

where *observed value* may be a magnitude or any quantity obtained from a certain scientific activity, and *error* may be a measurement error. We are interested in reducing the variability of the observed values by possibly minimizing the errors. One way to this is by utilizing auxiliary information based on some predictor variables X_i. In parametric regression, we model the relationship between the observed value

and the predictor or regressor variables by a certain functional (known) form; most commonly, we assume that the functional form of the relationship is a lower order polynomial, say a linear function, i.e.,

$$Y = \beta_o + \beta_1 X + \epsilon. \tag{1.2}$$

This approach has various drawbacks; there are many instances where the relationship cannot be accurately summarized by a polynomial of certain degree. Hence, the usual regression procedure may not provide a flexible tool for accurately capturing the trend of the true relationship at all times. We discuss some of the details regarding the differences between parametric and nonparametric regression methods in Chapter 2.

Similarly to regression, nonparametric regression describes the expected mean value of a dependent variable as a function of other variables (regressors) but with minimal assumptions about the functional form of the relationship. For instance, we may only assume that the relationship is governed by some smooth function f, i.e.,

$$Y = f(X) + \epsilon. \tag{1.3}$$

This is the origin of the term "smoothing"; in general, it is a procedure for summarizing the dependence of a response variable on a set of predictor variables. A corresponding estimator is often called "smoother"; it produces an estimate called "smooth".

There are two main uses of a smoother. Firstly, it is used to enhance (by reduction of noise or error) the visual appearance of the underlying trend or dependence. Secondly, it provides prediction of the true functional form of the relationship. The applications of smoothing range from arts, through politics to almost all scientific fields. For instance, smoothing has been used to forecast thunderstorm and earthquake patterns. It has also been used to characterize annual cycles of ambient ozone

concentrations and their correlation with meteorological conditions. In geography, smoothing may be used to smooth surfaces of elevations or to model prices of land (see Koenker and Mizera, 2001); and it has been used to provide better information on terrains, precipitation, temperature and analyze acid contents in water bodies like lakes and rivers (see Gu, 1991). In social sciences, smoothing has been used to determine economic indicators such as poverty thresholds and employment and unemployment rates. In geostatistics, a related technique to smoothing is "*kriging*", used for estimating the mean of spatial processes.

A relatively new application of smoothing is in the field of "data mining". Nowadays, the size of databases stored in computers is becoming larger and larger. This poses a problem to scientists: how to extract and make implicit information from extremely large databases useful to their end users. Data mining experts are challenged to come up with a technology that will analyze and visualize very large databases with a high level of accuracy. In this context, FEM is also helpful.

At present, there are several approaches for extracting and exploring valuable information from data sets. These approaches include local polynomial fitting, wavelet shrinkage method, Fourier method and roughness penalty method. All these smoothing methods have their pros and cons, depending on factors like availability of the software package, interpretability of smooths, statistical efficiency and quick computability. While statisticians espouse their personal favorites and argue about minor technical differences of these methods, the users and general public should not lose sight of the main idea that smoothing is a very useful tool and that there are effective ways to do it. In the univariate case, the independent variable is often time or some similar quantity that can be represented by a line. In the bivariate case, the covariates now form a plane, where each pair of covariate values is a point or location

in the plane. Recently, a number of univariate smoothing methods are already made available for its users. These are mostly integrated in many statistical software packages. Although univariate smoothing is important, it is also natural to ask whether these methods can be extended to the bivariate case, to smooth observations on a plane. Indeed, there are many instances where a valuable information depends on two regressor (predictor) variables. This necessitates the extension of univariate methods into a bivariate setting which is not straighforward. To my knowledge, only local polynomial and roughness penalty methods have natural bivariate extensions. Even for these two methods smoothing a surface over a plane is not easy, because not all the favorable properties of univariate smoothing hold true in a bivariate case. We discuss these properties in Section 2.3. A convenient property of bivariate smoothing is that we can still visualize the results; but there are additional complexities that need to be hurdled. As a direct consequence, few software packages have built-in bivariate smoothing procedures. This fact considerably motivated this thesis, focusing on the roughness penalty approach of bivariate smoothing. More specifically, we concentrate on the numerical solution of the penalized smoothing on bounded domains and on various variants of smoothing methods related to this problem.

In this objective, the thesis is related to recent literature. For instance, Ramsay (1999) uniquely formulated a bivariate generalization of the univariate L-spline smoothing problem and applied his procedure (FE L-spline) to analyze image data. More recently, a paper of Roberts et al. (2003) proposed a discrete approximation to the standard thin-plate spline smoothing problem, called the finite element thin-plate spline (TPSFEM) method. These are some of the techniques that claim to be highly capable of fitting surfaces from very large databases and therefore play important roles in the data mining process.

The thesis implements TPSFEM and FE L-spline using simple linear functions as interpolating polynomials. We also propose an alternative bivariate smoothing method which is derived from the original L-spline smoothing problem and is called the modified FE L-spline (MFE L-spline) method. We implement and propose the generalized FE-based smoothing procedures in a bivariate setting, providing the corresponding Matlab codes and Noweb documentation which can be used by practitioners. All methods are tested on synthetic data and on practical examples involving spatial components and originally from meteorology and mining geology. Insights on possible localization properties of FEM, which are helpful in capturing the local features of the data are presented. Generalized cross-validation (GCV) method is also used for smoothing parameter λ selection.

Chapter 2

Principles of Smoothing via Roughness Penalty Approach

2.1 (Parametric) Regression Approaches

2.1.1 (Simple) Linear Regression

Linear regression is one of the most useful and widely used data analysis techniques. In its simplest form, assume that data pairs $(X_i, Y_i), i = 1, \ldots, n$ are given; we then describe their relationship by a model of the form

$$Y = \alpha + \beta X + \epsilon, \tag{2.1}$$

which is usually done by fitting (2.1) to the observed data. The parameters α and β are then estimated by the least squares method. In practice, if we find the dependence of Y on X not linear then we do not summarize it with a straight line. Instead, we

modify model (2.1) by adding terms such as X^2 hoping that it improves the fit of the model. In general, we improve the fit of the model by guessing the functional form based on what is seen from the data, which is often times difficult. Consequently, this leads us to consider other approaches, which in most cases will be more flexible than fitting model (2.1).

2.1.2 Polynomial Regression

There are many instances where the relationship between Y and X is far from being linear. Hence, we might consider generalizing model (2.1) and rewrite it as

$$Y = f(X) + \epsilon \tag{2.2}$$

where $f(X)$ is usually a lower order polynomial. This technique is widely used and is implemented just like multiple regression. Unfortunately this method has similar drawbacks. For the explanation and enumeration of some of these, the reader may consult Green and Silverman (1994).

2.2 Spline Smoothing

2.2.1 Spline Functions in Smoothing

The name "*spline*" was introduced into the smoothing literature by Schoenberg (1964). He observed that curves were drawn by shipbuilders using a tool called a mechanical spline that consisted of weights connected to a flexible strip. The classical definition of a spline function on $[a, b]$ given n knots $a \leq x_1 \leq x_2, \ldots, x_n \leq b$,

is defined as a polynomial in each of the intervals $[a, x_1], \ldots, (x_j, x_{j+1}), \ldots, (x_n, b]$. These piecewise polynomials are assumed to be joined at the knots to make the function continuous. They possess some specified number of continuous derivatives and also satisfy certain boundary conditions. For a comprehensive discussion on splines as piecewise polynomials, see de Boor (1978). One very useful and widely known spline function in the field of smoothing is the cubic spline which usually takes the form:

$$f(x) = \gamma_i(x - x_i)^3 + \tau_i(x - x_i)^2 + \alpha_i(x - x_i) + \omega_i \tag{2.3}$$

for every $x_i \leq x \leq x_{i+1}$ and for given constants $\gamma_i, \tau_i, \alpha_i, \omega_i, i = 1, \ldots, n$. We can define $x_o = a$ and $x_{n+1} = b$. Furthermore, the cubic spline is a function which possesses the following properties : i) it interpolates, i.e., $f(x) = y_i$, for all i; ii) it is smooth, i.e., $f(x), f'(x), f''(x)$ exist and, iii) it is a cubic polynomial on each $[x_i, x_{i+1}]$.

A more interesting spline in the literature is the natural cubic spline. A cubic spline on $[a, b]$ is said to be a natural cubic spline if its second and third derivatives are zero at a and b. These are the natural boundary conditions that imply that $\gamma_o = \tau_o = \gamma_n = \tau_n = 0$, i.e., f is linear on the two extreme intervals $[a, x_1]$ and $[x_n, b]$. The natural cubic spline f also satisfies the following inequality:

$$\int_a^b \{f''(x)\}^2 \, dx \leq \int_a^b \{g''(x)\}^2 \, dx, \tag{2.4}$$

for all $g \in \Lambda =$ { g : square integrable second derivative } where $f''(a) = f''(b) = 0$. That is, it attains the minimum energy possible among all functions that have square integrable second derivatives. The proof is done using integration by parts.

2.2.2 Univariate Spline Smoothing

The univariate smoothing spline is the solution to the following minimization problem. Given the regressor variable values $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ which satisfy $a < x_1 < x_2 < \ldots < x_n < b$ and observed values $\mathbf{y} = (y_1, y_2, \ldots, y_n)$, find f_λ from an appropriately defined collection of functions to minimize the *penalized least squares*

$$\sum_{i=1}^{n}(w_i)\{y_i - f(x_i)\}^2 + \lambda \int_a^b \{f^{(m)}(x)\}^2 dx. \tag{2.5}$$

The unique solution to this problem is a piecewise polynomial that has $2m-2$ continuous derivatives and which satifies the boundary conditions $f^{(k)}(b) = f^{(k)}(a) = 0$ for $k = m, m+1, \ldots, 2m-1$, see Wahba (1990).

In particular, for $m = 2$, we have the following smoothing problem. Let $x_1, x_2, \ldots, x_n$ in $[a, b]$ satisfy $a < x_1 < x_2 < \ldots < x_n < b$ with corresponding observations $y_1, y_2, \ldots, y_n$. Find a function f with smoothing parameter $\lambda \in \mathbb{R}^+$ that minimizes

$$S_\lambda(f) = \sum_{i=1}^{n}(w_i)\{y_i - f(x_i)\}^2 + \lambda \int_a^b \{f''(x)\}^2 dx \tag{2.6}$$

over all functions which have square integrable second derivatives where w_i's are real-valued weight functions. This is the cubic spline smoothing problem with solution f_λ, a natural cubic spline with knots at the points x_i. Wahba (2000) details some important features of natural cubic splines.

2.2.3 Minimizer of the Penalized Least Squares

If we minimize (2.5) then the minimizer $\hat{f}_\lambda$ is a spline of degree $2m-1$. Thus, there exists a basis $\phi_1, \phi_2, \ldots, \phi_n$ such that $\hat{f}_\lambda = \sum_{i=1}^{n} \xi_j \phi_j(x)$ for some ξ_j. For simplicity,

we assume that the weights $w_i = 1$, for all i, so we want to minimize

$$\sum_{i=1}^{n}\{y_i - f(x_i)\}^2 + \lambda\int_a^b \{f^{(m)}(x)\}^2 dx \tag{2.7}$$

$$= \sum_{i=1}^{n}\{y_i - \sum_{i=1}^{n}\xi_j\phi_j(x_i)\}^2 + \lambda\sum_{\forall ij}\xi_i\xi_j\int_a^b \phi_i{}^{(m)}(x)\phi_j{}^{(m)}(x)dx \tag{2.8}$$

$$= \| \mathbf{Y} - \mathbf{\Phi\Xi} \|^2 + \lambda\mathbf{\Xi}'P\mathbf{\Xi} \tag{2.9}$$

where $\mathbf{\Phi}_{ij} = \phi_j(x_i)$ and $P_{ij} = \int_a^b \phi_i{}^{(m)}(x)\phi_j{}^{(m)}(x)dx$. Hence, by the theory of generalized regression,

$$\hat{\mathbf{\Xi}} = [\mathbf{\Phi}'\mathbf{\Phi} + \lambda P]^{-1}\mathbf{\Phi}'\mathbf{Y} \tag{2.10}$$

and

$$\hat{\mathbf{Y}} = \mathbf{\Phi}\hat{\mathbf{\Xi}} = \mathbf{\Phi}[\mathbf{\Phi}'\mathbf{\Phi} + \lambda P]^{-1}\mathbf{\Phi}'\mathbf{Y} = \mathbf{\Theta}_\lambda\mathbf{Y}. \tag{2.11}$$

We may note that (2.7) is a linear combination of the weighted residual sum of squares $\sum_{i=1}^{n}(w_i)\{y_i - f(x_i)\}^2$ and the roughness penalty $\lambda\int_a^b\{f^{(m)}(x)\}^2dx$ for some w_i and $\lambda \in \mathbb{R}^+$. In this context we speak about "penalized least squares". The residual sum of squares is a common measure for the goodness-of-fit, or the closeness of $\hat{f}_\lambda$ to the data, while the roughness penalty is used to quantify the local variation or the smoothness of $\hat{f}_\lambda$. The smoothing paramater λ controls the trade-off between smoothness of the estimate and the fidelity of the data, i.e., the tradeoff between bias and variance. If $\lambda = 0$, the smoothing spline f_λ interpolates the data and therefore estimates the true f with small bias but with likely large variance. But as λ increases to infinity, $\hat{f}_\lambda$ becomes the least squares straight line, which may have low variance but very large bias. In practice, λ is often selected in an automatic way, typically by some cross-validation method; see Wahba (1990).

2.3 Extensions of Penalized Least Squares

2.3.1 From Univariate to Bivariate Smoothing

Thin-Plate Spline Surface Fitting

A natural bivariate smoother will be a solution to the following penalized least squares problem

$$S_\lambda\{f(\mathbf{x})\} = \sum_{i=1}^{n}(w_i)\{y_i - f(\mathbf{x}_i)\}^2 + \lambda J(f). \tag{2.12}$$

However, the transition from the univariate to bivariate case is not straighforward. It is not obvious how the univariate penalized regression, in particular the roughness penalty, can be generalized to the bivariate case. One possible aspect that may be seriously considered is the independence of the choice of the coordinate system: when one rotates and/or translates the data, one would like the form of the penalized least squares to remain the same.

Thus, a natural generalization of the roughness penalty may have the form

$$J(f) = \int\int_\Omega \left(\frac{\partial^2 f}{\partial {x_1}^2}\right)^2 + 2\left(\frac{\partial^2 f}{\partial x_1 \partial x_2}\right)^2 + \left(\frac{\partial^2 f}{\partial {x_2}^2}\right)^2 dx_1 dx_2. \tag{2.13}$$

The smooths obtained by solving (2.12) and employing (2.13) as the roughness penalty are called thin-plate splines. The adjective "*thin-plate*" comes from the physical interpretation of the penalty as the bending energy for the elastic thin plate lifted in the direction of the data points. A thin-plate spline is a smooth function which minimizes (2.13) and is the bivariate analogue of the cubic spline in univariate smoothing. In order to obtain closed-form solutions of (2.12), one has, however, to extend the domain of integration of the penalty to the whole plane, i.e., $\mathbb{R}^2$ where $\mathbb{R}$ is the set of all real numbers.

A function $f(\mathbf{x})$ is a thin-plate spline on the data set $\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x_n} \in \mathbb{R}^2$ if and only if $f(\mathbf{x})$ is of the form

$$f(\mathbf{x}) = \sum_{j=1}^{3} \nu_j \phi_j(\mathbf{x}_i) + \sum_{i=1}^{n} \delta_i U(\| \mathbf{x} - \mathbf{x}_i \|) \tag{2.14}$$

for suitable constants ν_j and δ_i where $U(r) = r^2 \log r^2$, $r > 0$. These constants or coefficients are determined by solving a linear system.

Some details on the construction of smooths are in Green and Silverman (1994, chap. 7); the improved computational algorithms can be found in Sibson and Stone (1991). Bivariate smoothing formulated in this way may be computationally expensive. The linear system that arises from univariate spline smoothing involves a sparse matrix, in contrast with thin-plate spline smoothing which involves a full matrix. This creates a computational disadvantage for the thin-plate method especially if the data size is very large. One of the motivations for *finite element method* (FEM) is to achieve a sparse matrix for the linear system defining the solution; this may make the computation of smooths or predictions substantially faster. Another advantage of FEM is that it tends to smooth well on the boundary of the domain or data site compared with the classical thin-plate spline (see Green and Silverman, 1994). If we choose to minimize (2.12) over a region $\Omega \subset \mathbb{R}^2$ then the minimizer is called the "*finite window thin-plate spline*". This is even more attractive as opposed to the usual thin-plate spline because of its more realistic interpretation of a finite flat thin-plate. However, in this case one cannot use the closed form (2.14) of the solution and has to involve more numerical methods (see Green and Silverman, 1994). The thin-plate penalty functional can be generalized also to higher derivatives and to higher

dimensions (d) as

$$J_m^d(f) = \sum_{\alpha_1+\cdots+\alpha_d=m} \frac{m!}{\alpha_1!\cdots\alpha_d!} \int \cdots \int_{\mathbb{R}^d} \left(\frac{\partial^m f}{\partial x_1^{\alpha_1} \cdots \partial x_d^{\alpha_d}} \right)^2 dx_1 \cdots dx_d. \tag{2.15}$$

Wahba (1990) has more details on smoothing with (2.15) as the penalty functional.

2.3.2 Squared Laplacian Penalty for L-spline Surface Fitting

Another way to bivariate smoothing is to generalize L-spline approach. In univariate setting, L-spline is defined to be a natural generalization of polynomial spline smoothing. It only differs from (2.12) in the functional form of the penalty being used. In this method, roughness penalties depend on some linear differential operator L which is defined as

$$L = D^m + \sum_{j=0}^{m-1} \omega_j D^j \tag{2.16}$$

where D^j denotes the jth derivative operator and the ω_is are continuous real valued weight functions. This leads to a new minimization problem as follows:

$$S_\lambda\{f(\mathbf{x})\} = \sum_{i=1}^{n} (w_i)\{y_i - f(x_i)\}^2 + \lambda \int_a^b \{Lf(x)\}^2 dx. \tag{2.17}$$

If we let $L = D^2$ (second derivative) then (2.17) becomes

$$S_\lambda\{f(\mathbf{x})\} = \sum_{i=1}^{n} (w_i)\{y_i - f(x_i)\}^2 + \lambda \int_a^b \{f''(x))\}^2 dx. \tag{2.18}$$

A different approach (reproducing-kernel) including the $O(n)$ algorithm for solving the L-spline smoothing problem can be found in Ramsay and Heckmann (1996; 1997). Examples of differential operators for some parametric families with their corresponding bases can also be found in Heckmann and Ramsay (2000).

In measuring the roughness of functions in the plane, Duchamp and Stuetzle (2001) stressed the bivariate version of f'' which is the Hessian of the form

$$\mathbf{H}_f = \begin{bmatrix} f_{x_1x_1} & f_{x_1x_2} \\ f_{x_1x_2} & f_{x_2x_2} \end{bmatrix} \tag{2.19}$$

and that choosing a roughness measure to be rotationally invariant limits one's choice to functions of the trace of the power of the Hessian. For instance, letting $L = \frac{\partial}{\partial {x_1}^2} + \frac{\partial}{\partial {x_2}^2} = \Delta$ will yield the squared-Laplacian of f, which is easily seen to be a function of the trace of $\mathbf{H}_f$, i.e., $(tr(\mathbf{H}_f))^2 = (\Delta f)^2 = (f_{x_1x_1} + f_{x_2x_2})^2$. It can be found out that the standard thin-plate penalty is also a function of the trace of H_f, i.e., $tr(\mathbf{H}_f^2) = (f_{x_1x_1})^2 + 2(f_{x_1x_2})^2 + (f_{x_2x_2})^2$. These functions, albeit different, give similar global roughness measures as

$$\int_{\mathbb{R}^2} tr(\mathbf{H}_f^2) dx_1 dx_2 = \int_{\mathbb{R}^2} (tr(\mathbf{H}_f))^2 dx_1 dx_2, \tag{2.20}$$

as is also partly proven in the same paper. Thus, a bivariate L-spline version of the cubic smoothing spline's minimization functional can be expressed as

$$S_\lambda\{f(\mathbf{x})\} = \sum_{i=1}^{n} (w_i)\{y_i - f(\mathbf{x}_i)\}^2 + \lambda \int_\Omega \{\Delta f\}^2 d\mu, \tag{2.21}$$

where the penalty functional is the squared Laplacian.

Summarizing, we have two major methods for bivariate smoothing; they differ in the form of the penalty. Additional variations may be obtained by considering various domains of integration and also by use of different numerical methods.

Chapter 3

Smooth Approximation by Finite Element Method (FEM)

3.1 Function Spaces

3.1.1 Certain Hilbert Sobolev Spaces

In the following discussion, we introduce relevant concepts in a bivariate setting. We assume that $\mathbf{x} \in \mathbb{R}^2$.

$\mathcal{L}^2(\Omega)$

This is the space of functions defined on a closed and bounded real interval that are square Lebesgue integrable, i.e.,

$$\mathcal{L}^2(\Omega) = \{u(\mathbf{x}) : \int_\Omega u^2(\mathbf{x})\, d\mathbf{x} \leq \infty\}. \tag{3.1}$$

$\mathcal{H}^m(\Omega)$

The *mth*-order "weak" derivative of $u(\mathbf{x}) \in \mathcal{H}^o(\Omega)$ is a function $v(\mathbf{x})$ that satisfies

$$\int_\Omega v(\mathbf{x})\phi(\mathbf{x})d\mathbf{x} = (-1)^m \int_\Omega u(\mathbf{x})\, \phi^{(m)}(\mathbf{x})\, d\mathbf{x} \tag{3.2}$$

and vanishes on the boundary of Ω for all $\phi(\mathbf{x}) \in C^o(\Omega)$. If such a function exists then $v(\mathbf{x}) = u^{(m)}(\mathbf{x})$. This definition can now be used to define a specific Hilbert space, the Sobolev space $\mathcal{H}^m$.

Definition

For a nonnegative integer m and a domain $\Omega \in \mathbb{R}^n$, the Sobolev space $\mathcal{H}^m(\Omega)$ is defined as

$$\mathcal{H}^m(\Omega) = \{\, v(\mathbf{x}) : D^{|\alpha|}u \in \mathcal{L}^2(\Omega),\ |\alpha| \leq m\} \tag{3.3}$$

where

$$D^{|\alpha|} = \frac{\partial^{|\alpha|}u}{\partial {x_1}^{\alpha_1} \partial {x_2}^{\alpha_2} \ldots \partial {x_n}^{\alpha_n}} \tag{3.4}$$

and

$$|\alpha| = \alpha_1 + \alpha_2 + \ldots, + \alpha_n. \tag{3.5}$$

If the normal derivative is zero on the boundary $\partial\Omega$ then this space will be referred to as $\mathcal{H}_o^m$.

3.1.2 Usual Inner Products in $\mathcal{H}^m(\Omega)$ Spaces

The inner product in $\mathcal{H}^o(\Omega)$ is the same as $\mathcal{L}^2(\Omega)$:

$$(\, u(\mathbf{x}), v(\mathbf{x})\,)_{\mathcal{L}^2(\Omega)} = \int_\Omega u(\mathbf{x})v(\mathbf{x})\, d\mathbf{x}. \tag{3.6}$$

The inner product in $\mathcal{H}^m(\Omega)$ with m general is

$$(u(\mathbf{x}), v(\mathbf{x}))_{\mathcal{H}^m(\Omega)} = \int_\Omega \sum_{|\alpha| \leq m} |D^{|\alpha|} u|^2 \, d\mathbf{x}. \tag{3.7}$$

These spaces are widely used since they provide natural domains for solutions of finite element problems.

3.2 FEM via Galerkin Formulation

We illustrate the fundamental concepts of *finite element method* (FEM) on the well-known *Galerkin* method - a computational technique for approximating the solutions of partial differential equations. The technique solves an integral formulation of a *variational problem* defined over a certain domain. The domain is subsequently partitioned into subdomains called *finite elements* and the solution in the form of a simpler polynomial function is determined on each element. This partitioning is often referred to as *meshing*. In this thesis, we consider only the most widespread triangular meshing procedure called Delaunay triangulation - the domain is divided into triangles satisfying certain optimality property. Only *conforming finite element* methods will be used; that is, the finite element space (*weak*) will always be a subspace of the solution space (*strong*).

In the subsequent exposition, we closely follow the ideas expressed by Flaherty (2000) and Johnson (1987), since they provide a good rationale for the subject matter. We emphasize that the optimality properties of the procedure are well-documented in the related literature (see Johnson, 1987). As an initial example, consider a partial differential equation (PDE)

$$\mathbb{L}[u(\mathbf{x})] = f(\mathbf{x}) \tag{3.8}$$

where $u(\mathbf{x})$ is the solution and $\mathbb{L}$ is some linear functional. If we are lucky enough to get the closed-form solution(*strong*) in a direct way then we are done. Unfortunately, this is not always possible. Multiplying (3.8) by a *test* or a *weight* function $v(\mathbf{x})$ and integrating over the problem domain Ω, we can rewrite (3.8) as

$$\int_{\Omega} v(\mathbf{x})\, \mathbb{L}[\, u(\mathbf{x})\,]\, d\mathbf{x} = \int_{\Omega} v(\mathbf{x})\, f(\mathbf{x})\, d\mathbf{x}. \tag{3.9}$$

Equivalently, if we use the $\mathcal{L}^2(\Omega)$ inner product defined in (3.6) to represent the integral of a product of two functions, we have

$$\int_{\Omega} v(\mathbf{x})\, (\mathbb{L}[u(\mathbf{x})] - f(\mathbf{x})) = 0. \tag{3.10}$$

We can see that if the inner product vanishes then the solution of (3.10) is also a solution of (3.8) for all functions $v(\mathbf{x}) \in \mathcal{L}^2(\Omega)$. The method used in the fomulation of the integral (3.10) is called *method of weighted residuals* (MWR). It is apparent that the name of the method comes from the fact that

$$residual = \mathbb{L}[u(\mathbf{x})] - f(\mathbf{x}). \tag{3.11}$$

In one dimensional FEM, integration by parts is used to eliminate higher order derivative terms from the formulation. However, this no longer works for higher dimensional problems. Instead, a bivariate approach using Green's Theorem or the Divergence Theorem is often used. Eliminating higher order derivative terms will give us the following general expression of the equation

$$A(\, u(\mathbf{x}), v(\mathbf{x})\,) = (\, f(\mathbf{x})\,, v(\mathbf{x})\,). \tag{3.12}$$

We avoid the discussion of the different boundary conditions; good contributions to this topic can be found in numerous references. Expression (3.12) is now called the

weak formulation of the problem; "*weak*" in a sense that the solutions may have less continuity in contrast to the strong solutions of (3.8). A new expression is introduced in the left hand side of (3.12). It is a bilinear form; it is linear in both u and v. There are many other techniques for solving other types of PDE problems; (see Becker et al., 1981; 1983; 1983), for instance.

Now, to construct an approximate solution to (3.12), we specifically consider approximations $U(\mathbf{x})$ and $V(\mathbf{x})$ for $u(\mathbf{x})$ and $v(\mathbf{x})$, respectively, of the form

$$u(\mathbf{x}) \approx U(\mathbf{x}) = \sum_{i=1}^{n} \xi_i \phi_i(\mathbf{x}) \tag{3.13}$$

$$v(\mathbf{x}) \approx V(\mathbf{x}) = \sum_{i=1}^{n} \varphi_i \psi_i(\mathbf{x}) \tag{3.14}$$

where ϕ_i and ψ_i are preselected. Our goal is to determine the coefficients ξ_i and φ_i in order to produce a good approximation of $u(\mathbf{x})$ and $v(\mathbf{x})$. We call the approximation $U(\mathbf{x})$ the *trial* function and $V(\mathbf{x})$ the *test* function; they belong to a certain finite-dimensional space $V_h(\Omega)$, which is commonly called the *trial* or *test* function subspace.

We now summarize the *Galerkin* formulation as follows: Let V_h be any finite dimensional space whose basis belongs to a certain function space $\mathcal{V}(\Omega)$, i.e.,

$$V_h = span\{\, \psi_1(\mathbf{x}), \psi_2(\mathbf{x}), \ldots, \psi_n(\mathbf{x}) \,\} = \{\, v_h : v_h = \sum_{i=1}^{n} \varphi_i \, \psi_i \,\} \subset \mathcal{V}(\Omega) \tag{3.15}$$

where $\Omega \subset \mathbb{R}^2$. The Galerkin approximate weak solution is

$$u_h(\mathbf{x}) = \sum_{i=1}^{n} \varphi_i \psi_i(\mathbf{x}). \tag{3.16}$$

Its coefficients are determined by solving the linear system of equations:

$$A(u_h(\mathbf{x}) \,,\, v_h(\mathbf{x})\,) = (\, f(\mathbf{x}) \,,\, v_h(\mathbf{x})\,) \tag{3.17}$$

for every $v_h(\mathbf{x}) \in V_h$. The following theorem is very useful in constructing (3.17).

Characterization Theorem. *Let F be a linear functional on a linear space V and suppose that $A(\cdot,\cdot)$ is a symmetric, positive definite linear form on V. Then the quantity*

$$A(u, u) = 2F(u) \tag{3.18}$$

attains its minimum if and only if

$$A(u, v) = F(v), \quad \textit{for all } v \textit{ in } V. \tag{3.19}$$

Consequently, (3.17) can be rewritten in matrix form as

$$\begin{bmatrix} A(\psi_1,\psi_1) & \dots & A(\psi_1,\psi_n) \\ A(\psi_2,\psi_1) & \dots & A(\psi_2,\psi_n) \\ \vdots & \vdots & \ddots \\ A(\psi_n,\psi_1) & \dots & A(\psi_n,\psi_n) \end{bmatrix} \begin{bmatrix} \varphi_1 \\ \varphi_2 \\ \vdots \\ \varphi_n \end{bmatrix} = \begin{bmatrix} (f,\psi_1) \\ (f,\psi_2) \\ \vdots \\ (f,\psi_n) \end{bmatrix}. \tag{3.20}$$

Solving (3.20) will give us the coefficients in (3.13).

3.2.1 The Basis

In this thesis, we only consider the simplest finite element basis, the *piecewise-linear polynomial* on triangular elements. It is the bivariate analogue of the "hat" function in univariate FEM. We assume a triangulation $\mathcal{N} = \{\eta_i\}_{i=1}^{d}$ of $\Omega \subset \mathbb{R}^2$ with n global nodes (vertices) corresponding to n data points in the x_1x_2 plane. Consider an arbitrary triangular element η_j with nodes (vertices) indexed as 1, 2, and 3 and coordinates $(x_{1_k}, x_{2_k}), k = 1, 2, 3$. The linear shape function on an element η_j associated with the global node j has the form

$$N_j(x_1, x_2) = a + bx_1 + cx_2, \text{ for all } (x_1, x_2) \text{ in } \eta_j, \tag{3.21}$$

where η_j is the triangular support of N_j. Imposing a condition that the shape function must be equal to 1 on the global node j (as local node r),

$$N_{r,e}(x_{1_k}, x_{2_k}) = 1\,,\ r = k\ \ ;\ r, k = 1, 2, 3 \tag{3.22}$$

and equal to 0 on the other two local nodes, we have the following linear system:

$$\begin{bmatrix} 1 & x_{1_k} & x_{2_k} \\ 1 & x_{1_l} & x_{2_l} \\ 1 & x_{1_m} & x_{2_m} \end{bmatrix} \begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \quad , k \neq l \text{ and } l \neq m\ ;\ k, l, m = 1, 2, 3. \tag{3.23}$$

Solving this system will determine the coefficients in (3.21). Figures 3.1 and 3.2 show sample shape and basis functions obtained from Flaherty (2000) and Johnson (1987). It is clear that basis functions are made up of shape functions from neighbouring elements. It may be noted that a basis function looks like a "tent" with no opening or hole.

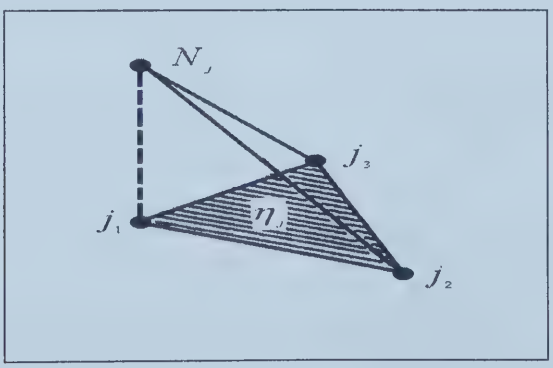

Figure 3.1: Linear Shape Function N_j for (local) Node 1 of Element η_j.

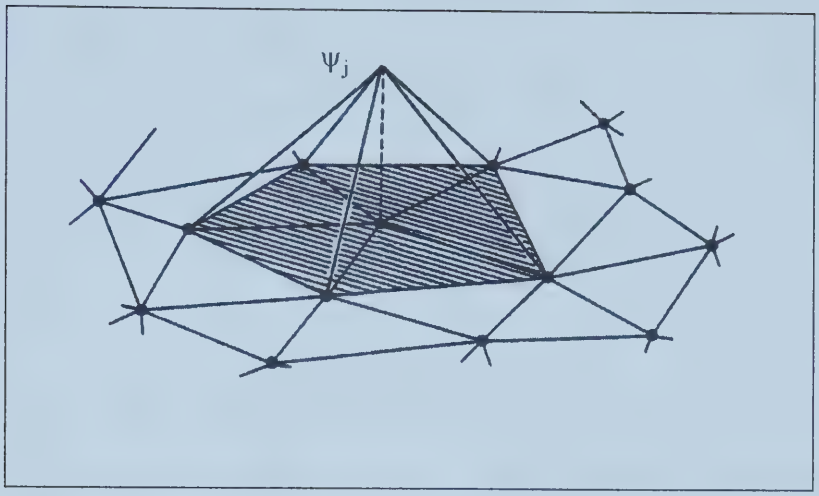

Figure 3.2: Basis Function ψ_j for a Cluster of Five Elements at Global Node j.

3.3 Finite Element(FE) Formulation of Bivariate Smoothing Problems

3.3.1 FE L-spline Formulation

Consider the bivariate L-spline smoothing problem in (2.17). Letting $L = \frac{\partial^2}{\partial x_1^2} + \frac{\partial^2}{\partial x_2^2} = \Delta$ and $w_i = 1$, we can rewrite (2.17) as

$$S_\lambda\{f(\mathbf{x})\} = \sum_{i=1}^{n}\{f(\mathbf{x}_i) - y_i\}^2 + \lambda\int_\Omega \{\Delta f\}^2 d\mu. \tag{3.24}$$

Ramsay (1999, chap. 4) formulated the bivariable generalization of the FE L-spline smoothing problem. His formulation used the characterization theorem. The following is the FE L-spline smoothing problem: Assuming a triangulation $\mathcal{N} = \{\eta_i\}_{i=1}^d$ of $\Omega \subset \mathbb{R}^2$, then the finite element approximation to the solution of (3.24) amounts to

finding $(f, g) \in \mathcal{H}^1_{h_o} \times \mathcal{H}^1_h$ such that

$$\begin{aligned} \lambda \int_\Omega \nabla g \cdot \nabla v - \sum_{i=1}^n f(\mathbf{x}_i)\, v(\mathbf{x}_i) &= -\sum_{i=1}^n v(\mathbf{x}_i)\, y_i, \quad \text{for all } v \text{ in } \mathcal{H}^1_{h_o}(\Omega) \\ \int_\Omega \nabla t \cdot \nabla f + \int_\Omega gt &= 0, \quad \text{for all } t \text{ in } \mathcal{H}^1_h(\Omega) \end{aligned} \tag{3.25}$$

where

$$g = \Delta f \ \text{ in } \Omega, \tag{3.26}$$

$\mathcal{H}^1_{h_o}$ is a subset of $\mathcal{H}^1_h$ consisting of functions with zero normal-derivative on the boundary $\delta\Omega$, ∇ is the gradient operator and " $\cdot$ " refers to scalar or dot product. The dot or scalar product of two vectors $\mathbf{a} = [a_1, \cdots, a_d]'$ and $\mathbf{b} = [b_1, \cdots, b_d]'$ is defined to be $\mathbf{a} \cdot \mathbf{b} = a_1 b_1 + a_2 b_2 + \cdots + a_d b_d$. For the complete derivation of formulation (3.25), see Ramsay (1999, chap. 4).

3.3.2 FE Thin-Plate Spline (TPSFEM) Formulation

Roberts et al. (2003) proposed a new method which they viewed as a discrete thin-plate spline. Their method combines favorable properties of finite element surface fitting with the standard thin-plate spline. They minimized the following functional:

$$S_\lambda\{\boldsymbol{f}(\mathbf{x}), s(\mathbf{x})\} = \sum_{i=1}^n \{s(\mathbf{x}_i) - y_i\}^2 + \lambda |\boldsymbol{f}|^2_{\mathcal{H}^1_h(\Omega)^2} \tag{3.27}$$

over all $\boldsymbol{f} = (f_1, f_2) \in \mathcal{H}^1_h(\Omega)^2$ and $s \in \mathcal{H}^2_h(\Omega)$ such that

$$\nabla s = \boldsymbol{f}. \tag{3.28}$$

Equivalently, condition (3.28) can also be expressed as

$$(\nabla s, \nabla v)_{\mathcal{L}^2_h(\Omega)^2} = (\boldsymbol{f}, \nabla v)_{\mathcal{L}^2_h(\Omega)^2}, \quad \text{for all } v \text{ in } \mathcal{H}^1_h(\Omega) \tag{3.29}$$

where

$$(\mathbf{u}, \mathbf{w})_{\mathcal{L}_h^2(\Omega)^2} = (u_1, w_1)_{\mathcal{L}_h^2(\Omega)} + (u_2, w_2)_{\mathcal{L}_h^2(\Omega)}. \tag{3.30}$$

The penalty is defined as

$$|\boldsymbol{f}|^2_{\mathcal{H}_h^1(\Omega)^2} = (f_1\,,\, f_1)_{\mathcal{H}_h^1(\Omega)} + (f_2\,,\, f_2)_{\mathcal{H}_h^1(\Omega)} \tag{3.31}$$

where the semi-inner product is given by

$$(f_i\,,\, f_i)_{\mathcal{H}_h^1(\Omega)} = \int_\Omega \{\frac{\partial f_i}{\partial x_1}\frac{\partial f_i}{\partial x_1} + \frac{\partial f_i}{\partial x_2}\frac{\partial f_i}{\partial x_2}\}\, dx_1\, dx_2 \;\;,\; i = 1, 2. \tag{3.32}$$

3.4 Setting Up the Linear System

3.4.1 Bivariable FE L-spline Set-Up

At the beginning, we assume that there are n observed function values y_i at n data points $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$ where $\mathbf{x}_i = (x_{1_i}, x_{2_i}) \in \mathbb{R}^2$. We further assume that there are as many basis functions as there are data points. Let $\{\psi_i\}_{i=1}^n$ be the set of basis functions corresponding to a set of nodal variables (vertices). In particular, we consider each data point $\mathbf{x}_i$ in the plane as our nodal variable. We can now express the FE L-spline smoothing problem as finding $(\mathbf{f}, \mathbf{g}) \in \mathbb{R}^n \times \mathbb{R}^n$ such that

$$\lambda \int (\mathbf{v}^T \psi_{x_1} \psi_{x_1}^T \mathbf{g}) + \int (\mathbf{v}^T \psi_{x_2} \psi_{x_2}^T \mathbf{g}) - \mathbf{v}^T \mathbf{I}_n \mathbf{f} = -\mathbf{v}^T \mathbf{y}, \quad \text{for all } \mathbf{v} \text{ in } \mathbb{R}^n \tag{3.33}$$

and

$$\int (\mathbf{t}^T \psi_{x_1} \psi_{x_1}^T \mathbf{f}) + \int (\mathbf{t}^T \psi_{x_2} \psi_{x_2}^T \mathbf{f}) + \int (\mathbf{t}^T \psi \psi^T \mathbf{g}) = \mathbf{0}, \quad \text{for all } \mathbf{t} \text{ in } \mathbb{R}^n, \tag{3.34}$$

where $f = \psi^T\mathbf{f} = \mathbf{f}^T\psi, g = \psi^T\mathbf{g} = \mathbf{g}^T\psi, v = \psi^T\mathbf{v} = \mathbf{v}^T\psi$ and $t = \psi^T\mathbf{t} = \mathbf{t}^T\psi$. These equations can be further expressed as

$$\begin{aligned} \lambda \int (\psi_{x_1}\psi_{x_1}^T + \psi_{x_2}\psi_{x_2}^T)\mathbf{g} + \mathbf{I}_n\mathbf{f} &= \mathbf{y} \\ \int (\psi_{x_1}\psi_{x_1}^T + \psi_{x_2}\psi_{x_2}^T)\mathbf{f} - \int (\psi\psi^T)\mathbf{g} &= \mathbf{0}, \end{aligned} \tag{3.35}$$

in compact notation as

$$\begin{bmatrix} \lambda\mathbf{Q} & \mathbf{I}_n \\ \mathbf{P} & -\mathbf{Q}^\mathbf{T} \end{bmatrix} \begin{bmatrix} \mathbf{g} \\ \mathbf{f} \end{bmatrix} = \begin{bmatrix} \mathbf{y} \\ \mathbf{0} \end{bmatrix} \tag{3.36}$$

where $\mathbf{Q} = \int (\psi_{x_1}\psi_{x_1}^T + \psi_{x_2}\psi_{x_2}^T)$ and $\mathbf{P} = \int (\psi\psi^T)$. This linear system can be solved using a suitable linear algebra technique. For more details on the uniqueness of the FE L-spline solutions under different boundary constraints, see Ramsay (1999).

3.4.2 Bivariable TPSFEM Set-Up

Let $\psi(\mathbf{x}) = [\psi_1(\mathbf{x}), \psi_2(\mathbf{x}), \ldots, \psi_n(\mathbf{x})]^T$ be the vector of basis functions for the finite element space $\mathcal{H}_h^1$. Then correspondingly, we can express s and $\boldsymbol{f} = (f_1, f_2)$ as

$$s(\mathbf{x}) = \psi^T\mathbf{c} \ , \ f_1(\mathbf{x}) = \psi^T\mathbf{c_1} \text{ and } f_2(\mathbf{x}) = \psi^T\mathbf{c_2} \tag{3.37}$$

where $\mathbf{c}, \mathbf{c_1}, \mathbf{c_2} \in \mathbb{R}^n$ are the coefficients of the linear representations above. From (3.29), we have

$$\int_\Omega \nabla v \cdot \nabla s \, d\mu = \int_\Omega \nabla v \cdot \boldsymbol{f} \, d\mu \tag{3.38}$$

where " $\cdot$ " refers to dot or scalar product previously defined. Substituting (3.37) to (3.38), we get

$$\int_\Omega \nabla(\psi^T\mathbf{d}) \cdot \nabla(\psi^T\mathbf{c}) \, d\mu = \int_\Omega \nabla(\psi^T\mathbf{d}) \cdot (f_1, f_2) \, d\mu. \tag{3.39}$$

Simplifying (3.39), we get

$$\int_\Omega (\nabla\psi^T \cdot \nabla\psi)\,\mathbf{c}\; d\mu = \int_\Omega (\psi_{x_1}\,,\, \psi_{x_1}) \cdot (\psi^T\mathbf{c_1}\,,\, \psi^T\mathbf{c_2})\; d\mu. \tag{3.40}$$

Also, the right hand side of (3.40) can be rewritten as

$$\int_\Omega (\psi_{x_1}\psi^T\mathbf{c_1} + \psi_{x_2}\psi^T\mathbf{c_2})\; d\mu \tag{3.41}$$

where ψ_{x_i} is the partial derivative of $\psi(\mathbf{x})$ with respect to x_i. Finally, the relationship between s and $\boldsymbol{f}$ is given by

$$\mathbf{Qc} = \mathbf{C_1c_1} + \mathbf{C_2c_2} \tag{3.42}$$

where $\mathbf{Q} = \int_\Omega \nabla\psi^T \cdot \nabla\psi\; d\Omega$, $\mathbf{C_1} = \int_\Omega \psi_{x_1}\psi^T$ and $\mathbf{C_2} = \int_\Omega \psi_{x_2}\psi^T$. In addition, the penalty is defined in (3.31) as

$$|\boldsymbol{f}|^2_{\mathcal{H}^1_h(\Omega)^2} = (f_1\,,\, f_1)_{\mathcal{H}^1_h(\Omega)} + (f_2\,,\, f_2)_{\mathcal{H}^1(\Omega)}. \tag{3.43}$$

Substituting (3.37) to (3.43), we get

$$|\boldsymbol{f}|^2_{\mathcal{H}^1_h(\Omega)^2} = (\psi^T\mathbf{c_1}\,,\, \psi^T\mathbf{c_1})_{\mathcal{H}^1_h(\Omega)} + (\psi^T\mathbf{c_2}\,,\, \psi^T\mathbf{c_2})_{\mathcal{H}^1_h(\Omega)}, \tag{3.44}$$

$$= \int_\Omega ((\psi^T_{x_1}\mathbf{c_1}\psi^T_{x_1}\mathbf{c_1} + \psi^T_{x_2}\mathbf{c_1}\psi^T_{x_2}\mathbf{c_1}) + (\psi^T_{x_1}\mathbf{c_2}\psi^T_{x_2}\mathbf{c_2} + \psi^T_{x_2}\mathbf{c_2}\psi^T_{x_2}\mathbf{c_2}))\; d\mu, \tag{3.45}$$

$$= \int_\Omega ((\mathbf{c_1}^T\psi_{x_1}\psi^T_{x_1}\mathbf{c_1} + \mathbf{c_1}^T\psi_{x_2}\psi^T_{x_2}\mathbf{c_1}) + (\mathbf{c_2}^T\psi_{x_1}\psi^T_{x_2}\mathbf{c_2} + \mathbf{c_2}^T\psi_{x_2}\psi^T_{x_2}\mathbf{c_2}))\; d\mu, \tag{3.46}$$

$$= \mathbf{c_1}^T(\int_\Omega \nabla\psi^T \cdot \nabla\psi)\mathbf{c_1} + \mathbf{c_2}^T(\int_\Omega \nabla\psi^T \cdot \nabla\psi)\mathbf{c_2}. \tag{3.47}$$

This can be further simplified as

$$|\boldsymbol{f}|^2_{\mathcal{H}^1_h(\Omega)^2} = \mathbf{c_1}^T\mathbf{Qc_1} + \mathbf{c_2}^T\mathbf{Qc_2} \tag{3.48}$$

where $\mathbf{Q} = (\nabla\psi^T, \nabla\psi)_{\mathcal{L}^2(\Omega)}$ which is the same $\mathbf{Q}$ defined in (3.36). Thus, the TPSFEM smoothing problem consists of minimizing the following functional

$$S_{\lambda_h}\{\mathbf{c}, \mathbf{c_1}, \mathbf{c_2}\} = \sum_{i=1}^{n}\{\psi(\mathbf{x}_i)^T\mathbf{c} - y_i)\}^2 + \lambda(\mathbf{c_1}^T\mathbf{Q}\mathbf{c_1} + \mathbf{c_2}^T\mathbf{Q}\mathbf{c_2}) \tag{3.49}$$

over all $\mathbf{c}, \mathbf{c_1}, \mathbf{c_2} \in \mathbb{R}^n$ subject to the constraint

$$\Psi(\mathbf{c}, \mathbf{c_1}, \mathbf{c_2}) = \mathbf{Q}\mathbf{c} - \mathbf{C_1}\mathbf{c_1} - \mathbf{C_2}\mathbf{c_2} = \mathbf{0}. \tag{3.50}$$

Using Lagrange method, we have

$$\begin{aligned} \tfrac{\delta}{\delta\mathbf{c}}\{S_{\lambda_h}\{\mathbf{c}, \mathbf{c_1}, \mathbf{c_2}\} + 2\mathbf{l}^T\Psi(\mathbf{c}, \mathbf{c_1}, \mathbf{c_2})\} \quad = \quad & 2\textstyle\sum_{i=1}^{n}\psi(\mathbf{x}_i)\psi(\mathbf{x}_i)^T\mathbf{c} \\ & -2\textstyle\sum_{i=1}^{n}\psi(\mathbf{x}_i)y_i + 2\mathbf{Q}^T\mathbf{l} \end{aligned} \tag{3.51}$$

where $\psi(\mathbf{x}_i) = [\psi_1(\mathbf{x}_i), \psi_2(\mathbf{x}_i), \ldots, \psi_n(\mathbf{x}_i)]^T$. In more compact form, (3.51) can be written as

$$\mathbf{B}\mathbf{c} - \mathbf{b} + \mathbf{Q}^T\mathbf{l}, \tag{3.52}$$

where

$$\mathbf{B} = \sum_{i=1}^{n}\psi(\mathbf{x}_i)\psi(\mathbf{x}_i)^T \tag{3.53}$$

and

$$\mathbf{b} = \sum_{i=1}^{n}\psi(\mathbf{x}_i)y_i. \tag{3.54}$$

Likewise,

$$\frac{\delta}{\delta\mathbf{c_1}}\{S_{\lambda_h}\{\mathbf{c}, \mathbf{c_1}, \mathbf{c_2}\} + 2\mathbf{l}^T\Psi(\mathbf{c}, \mathbf{c_1}, \mathbf{c_2})\} = 2\lambda\mathbf{Q}\mathbf{c_1} - 2\mathbf{C_1}^T\mathbf{l} \tag{3.55}$$

and

$$\frac{\delta}{\delta\mathbf{c_2}}\{S_{\lambda_h}\{\mathbf{c}, \mathbf{c_1}, \mathbf{c_2}\} + 2\mathbf{l}^T\Psi(\mathbf{c}, \mathbf{c_1}, \mathbf{c_2})\} = 2\lambda\mathbf{Q}\mathbf{c_2} - 2\mathbf{C_2}^T\mathbf{l}. \tag{3.56}$$

Roberts et al. (2003) call this the discrete constrained minimization problem and is equivalent to solving the following linear system

$$\begin{bmatrix} \mathbf{B} & \mathbf{0} & \mathbf{0} & \mathbf{Q} \\ \mathbf{0} & \lambda\mathbf{Q} & \mathbf{0} & -\mathbf{C_1}^T \\ \mathbf{0} & \mathbf{0} & \lambda\mathbf{Q} & -\mathbf{C_2}^T \\ \mathbf{Q} & -\mathbf{C_1} & -\mathbf{C_2} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{c} \\ \mathbf{c_1} \\ \mathbf{c_2} \\ \mathbf{l} \end{bmatrix} = \begin{bmatrix} \mathbf{b} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix} \tag{3.57}$$

where $\mathbf{l}$ is the Lagrange multiplier.

3.5 Modified FE Formulation of the Bivariate L-spline Smoothing Problem

3.5.1 Modified FE L-spline (MFE L-spline) Formulation

In this section, we formulate a new bivariate L-spline smoothing procedure. Recall Ramsay's L-spline minimization functional (3.24) which is

$$S_\lambda\{f(\mathbf{x})\} = \sum_{i=1}^{n}\{f(\mathbf{x}_i) - y_i\}^2 + \lambda\int_\Omega \{\Delta f\}^2 \, d\mu. \tag{3.58}$$

If we let $g = \Delta f$ then

$$\int_\Omega g\,t = \int_\Omega (\Delta f)\,t \ , \ \text{for all } t \text{ in } \mathcal{H}^1(\Omega). \tag{3.59}$$

By Green's theorem, we have

$$\int_\Omega (\nabla f)\cdot(\nabla t) = \int_{\partial\Omega} t\frac{\partial f}{\partial \nu}\, dz - \int_\Omega (\Delta f)\,t \tag{3.60}$$

where $\frac{\partial f}{\partial \nu} = \frac{\partial f}{\partial x_1}\nu_1 + \frac{\partial f}{\partial x_2}\nu_2$. Using this theorem and condition that the normal derivative of f is zero on the boundary $\partial\Omega$, it can be seen that

$$\int_\Omega (\Delta f)\, t \; d\mu = -\int_\Omega (\nabla f) \cdot (\nabla t)\, d\mu, \tag{3.61}$$

where " $\cdot$ " is the scalar or dot product operation. Consequently, (3.58) can now be rewritten as

$$S_\lambda\{f(\mathbf{x}), g(\mathbf{x})\} = \sum_{i=1}^{n}\{f(\mathbf{x}_i) - y_i\}^2 - \lambda\int_\Omega \nabla f \cdot \nabla g \; d\mu. \tag{3.62}$$

This leads to the following variational problem: Find $(f, g) \in \mathcal{H}^1 \times \mathcal{H}^1$ such that

$$S_{\lambda_h}\{f(\mathbf{x}), g(\mathbf{x})\} = \sum_{i=1}^{n}\{f(\mathbf{x}_i) - y_i\}^2 - \lambda\int_\Omega \nabla f \cdot \nabla g \; d\mu \tag{3.63}$$

is minimum subject to the following constraint :

$$\int_\Omega \nabla t \cdot \nabla f + \int_\Omega g\, t \;=\; 0 \text{ , for all } t \text{ in } \mathcal{H}^1(\Omega). \tag{3.64}$$

Thus, a modified FE L-spline smoothing problem can be restated as follows : Find $(f, g) \in \mathcal{H}^1_{h_o} \times \mathcal{H}^1_h$ such that

$$S_{\lambda_h}\{f(\mathbf{x}), g(\mathbf{x})\} = \sum_{i=1}^{n}\{f(\mathbf{x}_i) - y_i\}^2 - \lambda\int_\Omega \nabla f \cdot \nabla g \; d\mu \tag{3.65}$$

is minimum subject to the following constraint :

$$\int_\Omega \nabla t \cdot \nabla f + \int_\Omega g\, t \;=\; 0 \text{ , for all } t \text{ in } \mathcal{H}^1_h(\Omega). \tag{3.66}$$

Note that the finite element solution consists of forming the triangulation and then finding the solution from the finite element space $\mathcal{H}^1_{h_o} \times \mathcal{H}^1_h$.

Note that this formulation differs from Ramsay only in the formulation of the minimization problem. We minimize the MFE L-spline minimization functional by not using the characterization theorem, which makes it less copmlicated.

3.5.2 Setting Up the Bivariable MFE L-spline's Linear System

We now construct the linear system for computing the MFE L-spline smooth. Consider the linear approximations in section 3.4.2, it can be shown that (3.62) is equivalent to

$$S_{\lambda_h}\{\mathbf{f},\mathbf{g},\mathbf{u}\} = \sum_{i=1}^{n}\{\psi(\mathbf{x}_i)^T\mathbf{f} - y_i\}^2 - \lambda\int_{\Omega}(\mathbf{g}^T\psi_{x_1}\psi_{x_1}{}^T\mathbf{f} + \mathbf{g}^T\psi_{x_2}\psi_{x_2}{}^T\mathbf{f})\ d\mu, \quad (3.67)$$

which is equal to

$$\sum_{i=1}^{n}\{\psi(\mathbf{x}_i)^T\mathbf{f} - y_i\}^2 - \lambda\mathbf{g}^T(\int_{\Omega}\nabla\psi^T\cdot\nabla\psi)\mathbf{f}\ d\mu. \quad (3.68)$$

Hence, the modified L-Spline smoothing problem is equivalent to minimizing

$$\sum_{i=1}^{n}\{\psi(\mathbf{x}_i)^T\mathbf{f} - y_i\}^2 - \lambda\mathbf{g}^T\mathbf{Q}\mathbf{f} \quad (3.69)$$

subject to

$$\int(\psi_{x_1}\psi_{x_1}^T + \psi_{x_2}\psi_{x_2}^T)\mathbf{f} + \int(\psi\psi^T)\mathbf{g} = \mathbf{0}, \quad (3.70)$$

which can be further simplified as

$$\Psi(\mathbf{f},\mathbf{g}) = \mathbf{Q}\mathbf{f} + \mathbf{P}\mathbf{g} = \mathbf{0}. \quad (3.71)$$

Again, applying Lagrangian technique, we have the following linear system:

$$\begin{bmatrix} 2\mathbf{B} & -\lambda\mathbf{Q}^T & 2\mathbf{Q}^T \\ -\lambda\mathbf{Q}^T & \mathbf{0} & 2\mathbf{P}^T \\ \mathbf{Q} & \mathbf{P} & \mathbf{0} \end{bmatrix}\begin{bmatrix} \mathbf{f} \\ \mathbf{g} \\ \mathbf{l} \end{bmatrix} = \begin{bmatrix} 2\mathbf{b} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix} \quad (3.72)$$

where $\mathbf{l}$ is the Lagrange multiplier. We emphasize that the matrix involved in the linear system (3.72) is larger than Ramsay's linear system (3.36). This makes a

little difference in the computational complexity involved in solving the linear system. However, we emphasize that the bivariate implementation of L-spline in this thesis can be made faster than Ramsay's original implementation. Nevertheless, the algorithms involved in both formulations are quite similar.

3.6 FE Formulation of Generalized Bivariate Smoothing Problems

3.6.1 The Construction of a Generalized Smooth : A Sample

We now present a generalization of the previous algorithms. Recall that the previous weight functions assume only a constant value 1. This can now be generalized in a sense that the weight functions can now assume two constant values, 0 and 1. Surprisingly, this assumption leads to a more flexible tool for smoothing surfaces especially if only a small data set is available. In the sequel, we only derive the generalization for the FE L-spline case as the others directly follow. But as we proceed, we will try to stress the important points to avoid subtleties in the discussion.

Subsequently, suppose that we have n original design points (with known response), n' is the number of design points (with unknown response values) we want to insert and $n + n' = N$. These unknown response values constitute a serious drawback in the implementation since the residual sum of squares will be undefined. However, assigning weights $w_i = 1$ for the n original known responses and $w_i = 0$ for the n' unknown responses (corresponding to new n' designed points in the plane) resolves the problem just stated. Fortunately, this procedure results in only a few

minor changes in our previous algorithms. It can also be found out that the independence of the penalty term from the response variable y_i makes the changes easier and simpler.

Consider now the L-spline minimization functional in (2.21),

$$S_\lambda\{f(\mathbf{x})\} = \sum_{i=1}^{N}(w_i)\{f(\mathbf{x}_i) - y_i\}^2 + \lambda\int_\Omega \{\Delta f\}^2 d\mu. \tag{3.73}$$

Note that the new design points will not contribute to the lack-of-fit (infidelity) of the data, if $w_i = 0$; however, we show later that these inserted design or data points will somehow help smooth surfaces or fits. Eventually, this will address our previous concern of producing more attractive and more informative fits of the data. Simplifying (3.73), we get

$$\sum_{i=1}^{N} w_i f(\mathbf{x}_i)^2 - 2\sum_{i=1}^{N} w_i f(\mathbf{x}_i) y_i + \sum_{i=1}^{N} w_i {y_i}^2 + \lambda\int_\Omega \{\Delta f\}^2 \, d\mu. \tag{3.74}$$

From the characterization theorem, we have

$$A(f,f) \; - \; 2F(f) = \sum_{i=1}^{N} w_i f(\mathbf{x}_i)^2 + \lambda\int_\Omega \{\Delta f\}^2 \, d\mu - 2\sum_{i=1}^{N} w_i f(\mathbf{x}_i) y_i. \tag{3.75}$$

We drop the third term in (3.74) since it is not a function of f. The theorem further implies that expression (3.75) will be minimum if $A(f,v) = F(v)$, i.e.,

$$\sum_{i=1}^{N} w_i f(\mathbf{x}_i) v(\mathbf{x}_i) + \lambda\int_\Omega (\Delta f)(\Delta v) \, d\mu = \sum_{i=1}^{N} w_i v(\mathbf{x}_i) y_i \;, \quad \text{for all } v \text{ in } \mathcal{H}_o^2. \tag{3.76}$$

If we let $g = \Delta f$ and by applying the Green's theorem (3.60) in the same fashion as in subsection (3.5.1), then we get the following variational problem : Find $(f,g) \in \mathcal{H}_o^1 \times \mathcal{H}^1$ which satisfy

$$g = \Delta f \Leftrightarrow \int_\Omega gt = \int_\Omega (\Delta f)t = -\int_\Omega (\nabla f)t \Leftrightarrow \int_\Omega gt = -\int_\Omega (\nabla f)t, \text{ for all } t \text{ in } \mathcal{H}^1 \tag{3.77}$$

and

$$\sum_{i=1}^{N} w_i f(\mathbf{x}_i) v(\mathbf{x}_i) + \lambda \int_{\Omega} (\nabla f) \cdot (\nabla v)\, d\mu = \sum_{i=1}^{N} w_i v(\mathbf{x}_i) y_i\,, \quad \text{for all } v \text{ in } \mathcal{H}_o^1. \tag{3.78}$$

Finally, we have the similar FE L-spline smoothing problem as follows : Find $(f, g) \in \mathcal{H}_{h_o}^1 \times \mathcal{H}_h^1$ such that

$$\begin{aligned} \lambda \textstyle\int_{\Omega} \nabla g \cdot \nabla v - \sum_{i=1}^{N} w_i f(\mathbf{x}_i)\, v(\mathbf{x}_i) &= -\sum_{i=1}^{N} w_i v(\mathbf{x}_i)\, y_i \quad \text{for all } v \text{ in } \mathcal{H}_{h_o}^1 \\ \textstyle\int_{\Omega} \nabla t \cdot \nabla f + \int_{\Omega} gt &= 0, \quad \text{for all } t \text{ in } \mathcal{H}_h^1 \end{aligned} \tag{3.79}$$

where w_i are the weight functions.

We now construct the linear system based on the newly assigned weights. From (3.79), we can now express the equivalent FE L-spline smoothing problem as finding $(\mathbf{f}, \mathbf{g}) \in \mathbb{R}^N \times \mathbb{R}^N$ satisfying

$$\lambda \int (\mathbf{v}^T \psi_{x_1} \psi_{x_1}^T \mathbf{g}) + \int (\mathbf{v}^T \psi_{x_2} \psi_{x_2}^T \mathbf{g}) - \mathbf{v}^T \mathbf{W}_N \mathbf{f} = -\mathbf{v}^T \mathbf{W}_N \mathbf{y}, \quad \text{for all } \mathbf{v} \text{ in } \mathbb{R}^N \tag{3.80}$$

and

$$\int (\mathbf{t}^T \psi_{x_1} \psi_{x_1}^T \mathbf{f}) + \int (\mathbf{t}^T \psi_{x_2} \psi_{x_2}^T \mathbf{f}) + \int (\mathbf{t}^T \psi \psi^T \mathbf{g}) = \mathbf{0}, \quad \text{for all } \mathbf{t} \text{ in } \mathbb{R}^N, \tag{3.81}$$

where $f = \psi^T \mathbf{f} = \mathbf{f}^T \psi, g = \psi^T \mathbf{g} = \mathbf{g}^T \psi, v = \psi^T \mathbf{v} = \mathbf{v}^T \psi, t = \psi^T \mathbf{t} = \mathbf{t}^T \psi$ and $\mathbf{W}_N$ (previously $\mathbf{I}_n$) is now an augmented diagonal matrix with 1 on the first n diagonal entries and 0 on the remaining n' diagonal entries. These equations can be further expressed as

$$\begin{aligned} \lambda \textstyle\int (\psi_{x_1} \psi_{x_1}^T + \psi_{x_2} \psi_{x_2}^T) \mathbf{g} + \mathbf{W}_N \mathbf{f} &= \mathbf{W}_N \mathbf{y} \\ \textstyle\int (\psi_{x_1} \psi_{x_1}^T + \psi_{x_2} \psi_{x_2}^T) \mathbf{f} - \int (\psi \psi^T) \mathbf{g} &= \mathbf{0}, \end{aligned} \tag{3.82}$$

which can be compactly written as

$$\begin{bmatrix} \lambda\mathbf{Q} & \mathbf{W}_N \\ \mathbf{P} & -\mathbf{Q}^{\mathrm{T}} \end{bmatrix} \begin{bmatrix} \mathbf{g} \\ \mathbf{f} \end{bmatrix} = \begin{bmatrix} \mathbf{y}' \\ \mathbf{0} \end{bmatrix} \tag{3.83}$$

where $\mathbf{Q}$ and $\mathbf{P}$ are defined similarly while $\mathbf{y}'$ is now the new response vector of length $n + n' = N$. This new response vector $\mathbf{y}'$ can be obtained by using the first n entries of the old or original response vector and 0 on the remaining n' entries. This is still a linear system which can be solved using a similar technique used previously. Solving this linear system will give a better smooth $\mathbf{f}$.

As a result of the new weighting scheme, only few changes can be made to the TPSFEM and MFE L-spline algorithms. Firstly, recall the matrix $\mathbf{B}$ (previously an identity matrix $\mathbf{I}_n$) used in TPSFEM and MFE L-spline methods. This matrix is now the $\mathbf{W}_N$ matrix just defined earlier. Lastly, the former vector $\mathbf{b}$ is now the new response vector $\mathbf{y}'$. It is also emphasized that the size of the matrices involved in the linear system are increased correspondingly due to the n' data points added. However, the added points make FE-based bivariate smoothing methods more flexible in producing more appealing smooths.

Chapter 4

Implementation, Testing and Interpretation of Results

4.1 The Assembly

In the implementation process, a simpler approach for constructing the matrices is introduced. This approach stems from the formulae of Zienkiewicz and Taylor (2000), which we present in the following theorem:

Theorem. *Let a triangle be defined in the x_1x_2 plane by three points (x_{1k}, x_{2k}), (x_{1l}, x_{2l}), (x_{1m}, x_{2m}) with the origin at the centroid, i.e., $\bar{x_1} = \frac{x_{1k}+x_{1l}+x_{1m}}{3} = \bar{x_2} = \frac{x_{2k}+x_{2l}+x_{2m}}{3} = 0$. Integrating over the triangle area, we get:*

$$\int x_1 dx_1 dx_2 = \int x_2\, dx_1 dx_2 = 0, \tag{4.1}$$

$$\int {x_1}^2 dx_1 dx_2 = \frac{|\triangle|}{12}({x_{1k}}^2 + {x_{1l}}^2 + {x_{1m}}^2), \tag{4.2}$$

$$\int {x_2}^2 dx_1 dx_2 = \frac{|\triangle|}{12}({x_{2k}}^2 + {x_{2l}}^2 + {x_{2m}}^2), \tag{4.3}$$

$$\int x_1 x_2 dx_1 dx_2 = \frac{|\triangle|}{12}(x_{1k}x_{2k} + x_{1l}x_{2l} + x_{1m}x_{2m}), \tag{4.4}$$

where $|\triangle|$ is the area of the triangle element.

The area of the triangle can be computed as

$$|\triangle| = \frac{\left| det \begin{bmatrix} 1 & x_{1_k} & x_{2_k} \\ 1 & x_{1_l} & x_{2_l} \\ 1 & x_{1_m} & x_{2_m} \end{bmatrix} \right|}{2}, \tag{4.5}$$

where $x_{1k}, x_{2k}, x_{1l}, x_{2l}, x_{1m}, x_{2m}$ are the nodal component values.

Recall the matrices $\mathbf{P}$ and $\mathbf{Q}$,

$$\mathbf{P} = \int (\psi\psi^T) \text{ and } \mathbf{Q} = \int (\psi_{x_2}\psi_{x_1}^T + \psi_{x_2}\psi_{x_2}^T). \tag{4.6}$$

From Figure 3.2, we see that these bases are implicitly constructed by combining shape functions, i.e.,

$$\psi_j = \bigcup_{\eta_e} N_j(x_1, x_2). \tag{4.7}$$

Now, the ijth entry of $\mathbf{P}$ is calculated as

$$\mathbf{P}_{ij} = \int \psi_i \psi_j = \sum_{\eta_e} N_i(x_1, x_2) N_j(x_1, x_2) |\triangle_e|, \tag{4.8}$$

where summation is done over all the triangular elements. Letting

$$N_i(x_1, x_2) = a_i + b_i x_2 + c_i x_2 \tag{4.9}$$

and

$$N_j(x_1, x_2) = a_j + b_j x_1 + c_j x_2, \tag{4.10}$$

we get

$$\mathbf{P}_{ij} = \sum_{\eta_e} (a_i + b_i x_2 + c_i x_2)(a_j + b_j x_1 + c_j x_2)|\triangle_e|. \tag{4.11}$$

It may be noted that these shape functions vanish outside the support of the finite element basis. This makes the matrix sparse and makes FEM computationally more efficient. However, in order to use the previous theorem, we introduce a simple transformation. Let

$$u_1 = x_1 - \bar{x_1}, \text{ that is, } x_1 = u_1 + \bar{x_1} \tag{4.12}$$

and

$$u_2 = x_2 - \bar{x_2}, \text{ that is, } x_2 = u_2 + \bar{x_2}. \tag{4.13}$$

Substituting (4.12) and (4.13) to (4.11) will show that

$$\begin{aligned}
\mathbf{P}_{ij} \quad = \quad & \textstyle\sum_{\eta_e} \{ a_i a_j + (\ a_i b_j + b_i a_j\ )\bar{x_1} + (\ a_i c_j + c_i a_j\ )\bar{x_2} \\
& + (\ b_i c_j + b_j c_i\ )\{ \tfrac{(x_{1k}-\bar{x_1})(x_{2k}-\bar{x_2})+(x_{1l}-\bar{x_1})(x_{2l}-\bar{x_2})+(x_{1m}-\bar{x_1})(x_{2m}-\bar{x_2})}{12} \} \\
& + (\ b_i c_j + b_j c_i\ )(\ \bar{x_1}\bar{x_2}\ ) + (\ b_i b_j\ )\{ \tfrac{(x_{1k}-\bar{x_1})^2+(x_{1l}-\bar{x_1})^2+(x_{1m}-\bar{x_1}\ )^2}{12} \} \\
& + (b_i b_j)\bar{x_1}^2 + (c_i c_j)\bar{x_2}^2 + (c_i c_j)\{ \tfrac{(x_{2k}-\bar{x_2})^2+(x_{2l}-\bar{x_2})^2+(x_{2m}-\bar{x_2})^2}{12} \}\}|\triangle_e|
\end{aligned} \tag{4.14}$$

where $|\triangle_e|$ is the area of the triangle element η_e. Similarly, the ijth entry of $\mathbf{Q}$ is

$$\mathbf{Q}_{ij} \quad = \quad \int (\ \psi_{x_{1i}} \psi_{x_{1j}} + \psi_{x_{2i}} \psi_{x_{2j}}\ ), \tag{4.15}$$

which is the same as

$$\mathbf{Q}_{ij} = \sum_{\eta_e} (\ b_i b_j + c_i c_j\ )|\triangle_e|. \tag{4.16}$$

The general algorithms for constructing matrices $\mathbf{P}$ and $\mathbf{Q}$ can be found in Ramsay (1999). Moreover, the ijth entries of $\mathbf{C_1}$ and $\mathbf{C_2}$ are given by

$$\mathbf{C_{1}}_{ij} = \sum_{\eta_e} (\ b_i a_j + b_i b_j \bar{x_1} + b_i c_j \bar{x_2} \ ) |\triangle_e| \tag{4.17}$$

and

$$\mathbf{C_{2}}_{ij} = \sum_{\eta_e} (\ c_i a_j + c_i b_j \bar{x_1} + c_i c_j \bar{x_2} \ ) |\triangle_e|. \tag{4.18}$$

By construction, a basis function is unity at global node (vertex) j and zero outside the support of the basis function ψ_j. Thus, the matrices $\mathbf{B}$ and $\mathbf{b}$ are simply defined to be

$$\mathbf{B} = \mathbf{I}_n \quad \text{and} \quad \mathbf{b} = \mathbf{y}. \tag{4.19}$$

4.2 Applications

4.2.1 Synthetic Data: Wendelberger Function

In this case, we test FE-based methods by smoothing observations from a known function. In such a case we have an advantage of knowing how this function truly looks. Specifically, we smooth 200 values generated from the Wendelberger function, with noise added:

$$\begin{aligned} f(x_1, x_2) = 0.75exp(-((9x_1 - 2)^2 + \tfrac{(9x_2-2)^2)}{4}) \\ +0.75exp(-(\tfrac{(9x_1+1)^2}{49} + \tfrac{(9x_2+1)^2}{10})) \\ +0.5exp(-\tfrac{((9x_1-7)^2+(9x_2-3)^2)}{4}) \\ -0.2exp(-((9x_1 - 4)^2 + (9x_2 - 7)^2)) + \epsilon \end{aligned} \tag{4.20}$$

where x_1, x_2 are in $\mathbf{U}[-1,\ 1]$ and ϵ (error or noise) are independently and identically distributed with distribution $\mathbf{N}(0,\ (0.1)^2)$. Below is the original shape, without noise.

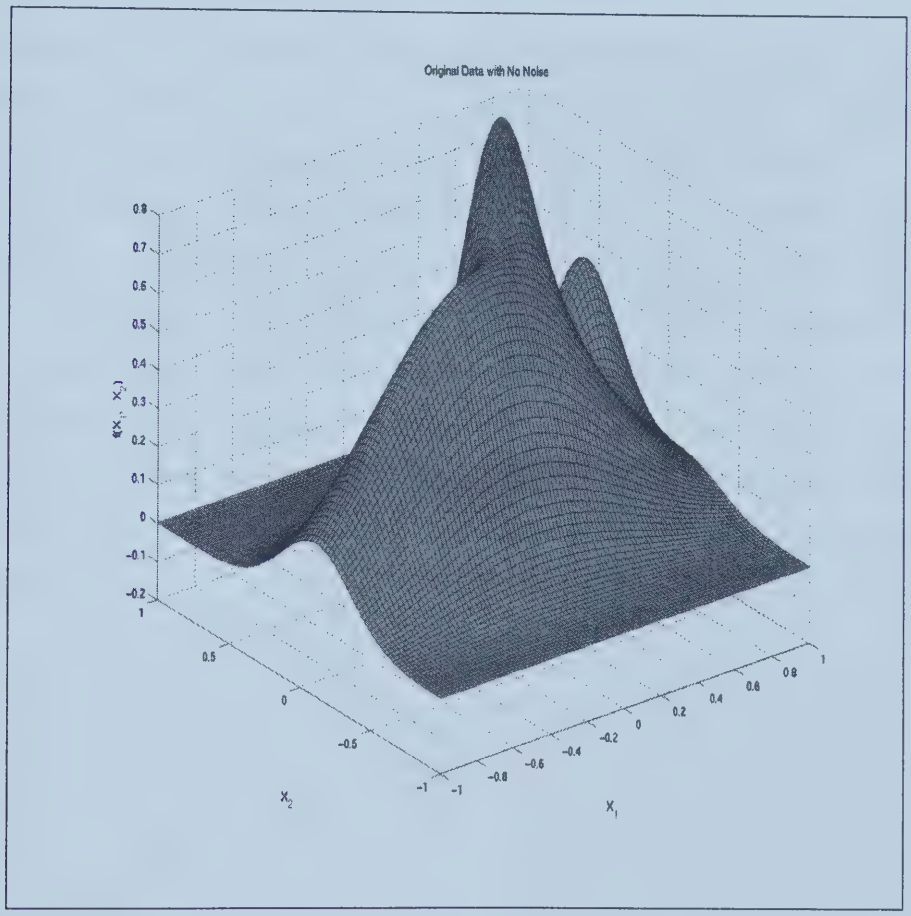

Figure 4.1: Graph of the Wendelberger Function.

We tested the FE-based methods by comparing the smooths with the standard thin-plate spline as a reference (benchmark). We use the triangulations generated only on the data points. As a result, piecewise linear smooths are somewhat "coarse"; nevertheless, such smooths are very useful for prediction.

Figures 4.2-4.5 below show the FE fits for the Wendelberger function together with that of standard thin-plate spline's. Apparently, the FE-based bivariate fits are quite similar to the standard thin-plate spline fit for certain values of λ. We see original observations (represented by the dots) that are getting farther away from the surface as the smoothing parameter value λ becomes large. These graphs also give the impression that the surface has only one peak. We also observe a perfect fit (interpolation) to the data for very small λ and a linear fit for very large λ. In these figures, we can only observe the general trends and structures of the regression fits as the smoothing parameter λ increases. But we could not see more informative and clearer structures of the fits. Hence, we later provide smoother versions of these surfaces.

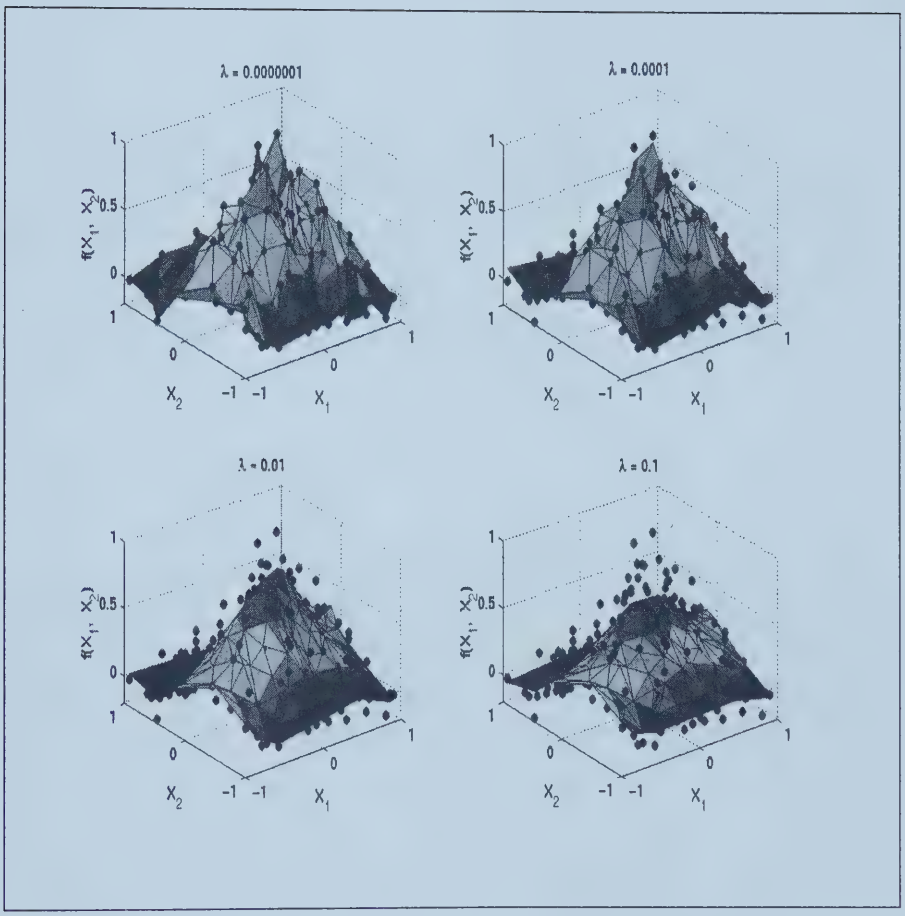

Figure 4.2: FE L-spline's Bivariate Fits of the Wendelberger Function as $\lambda \to \infty$.

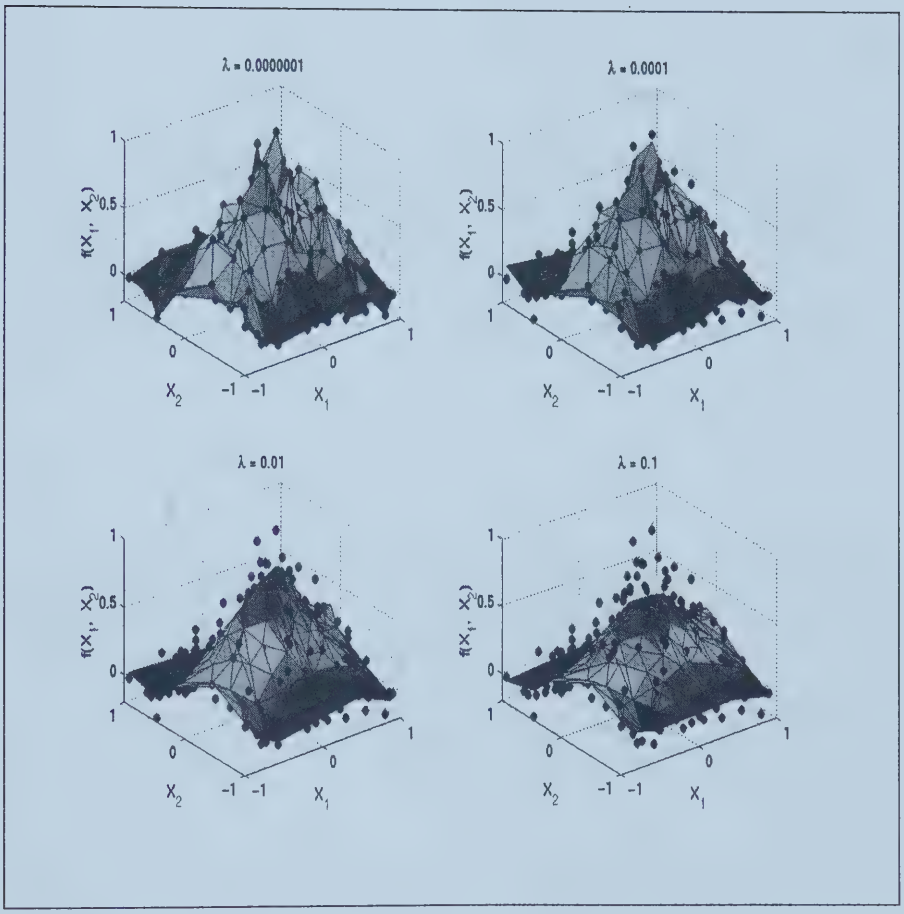

Figure 4.3: MFE L-spline's Bivariate Fits of the Wendelberger Function as $\lambda \to \infty$.

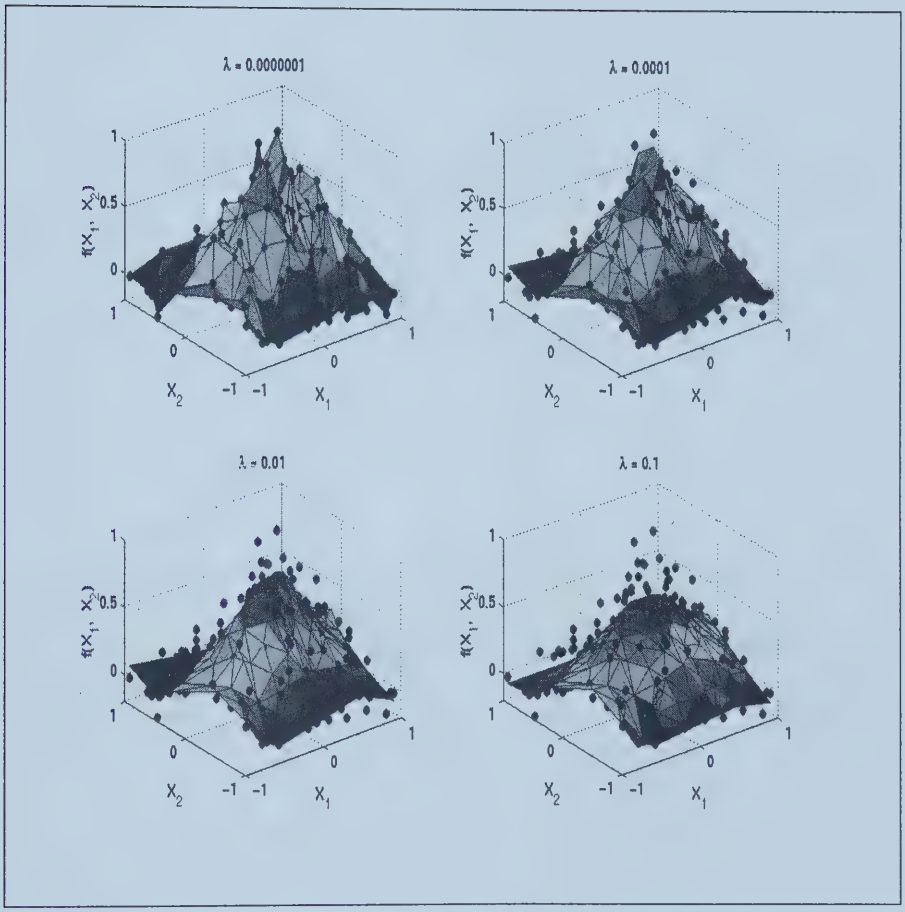

Figure 4.4: TPSFEM's Bivariate Fits of the Wendelberger Function as $\lambda \to \infty$.

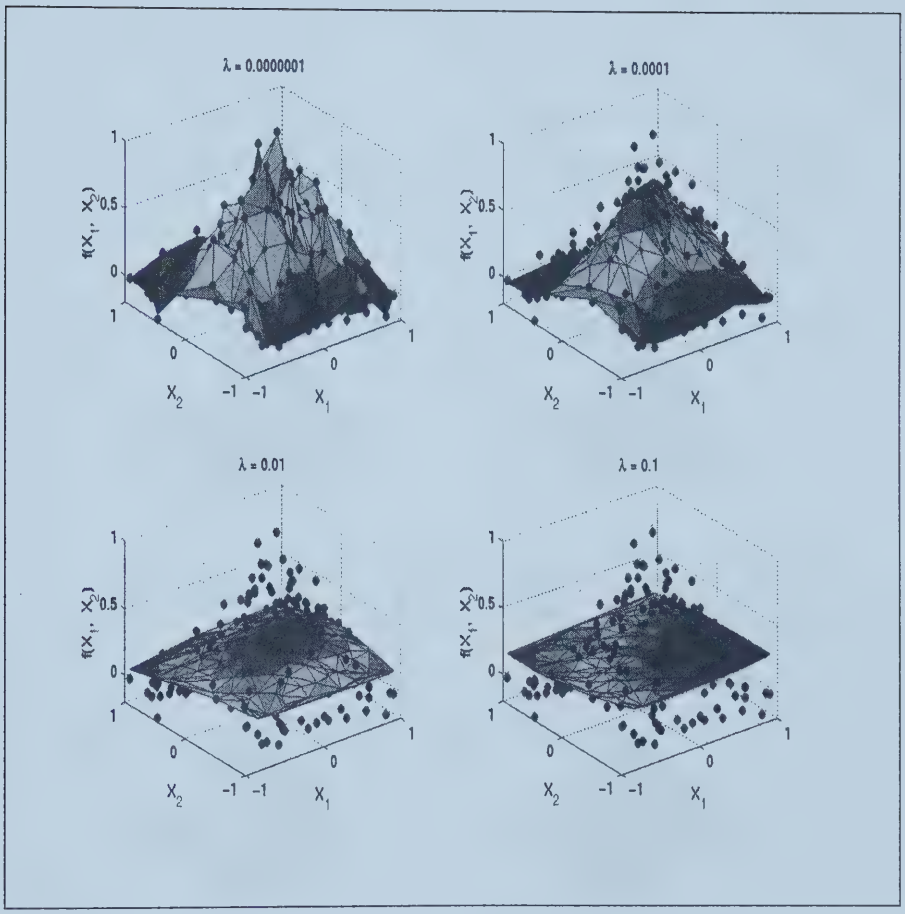

Figure 4.5: Thin-Plate Spline Fits of the Wendelberger Function as $\lambda \to \infty$.

4.2.2 Generalized Smooths for Synthetic Data: Wendelberger Function

Figures 4.6-4.9 are the resulting generalized FE smooths together with the thin-plate spline's fit for the synthetic data. Apparently, these methods now provide more informative and more attractive smooths, which can be generally helpful in practice.

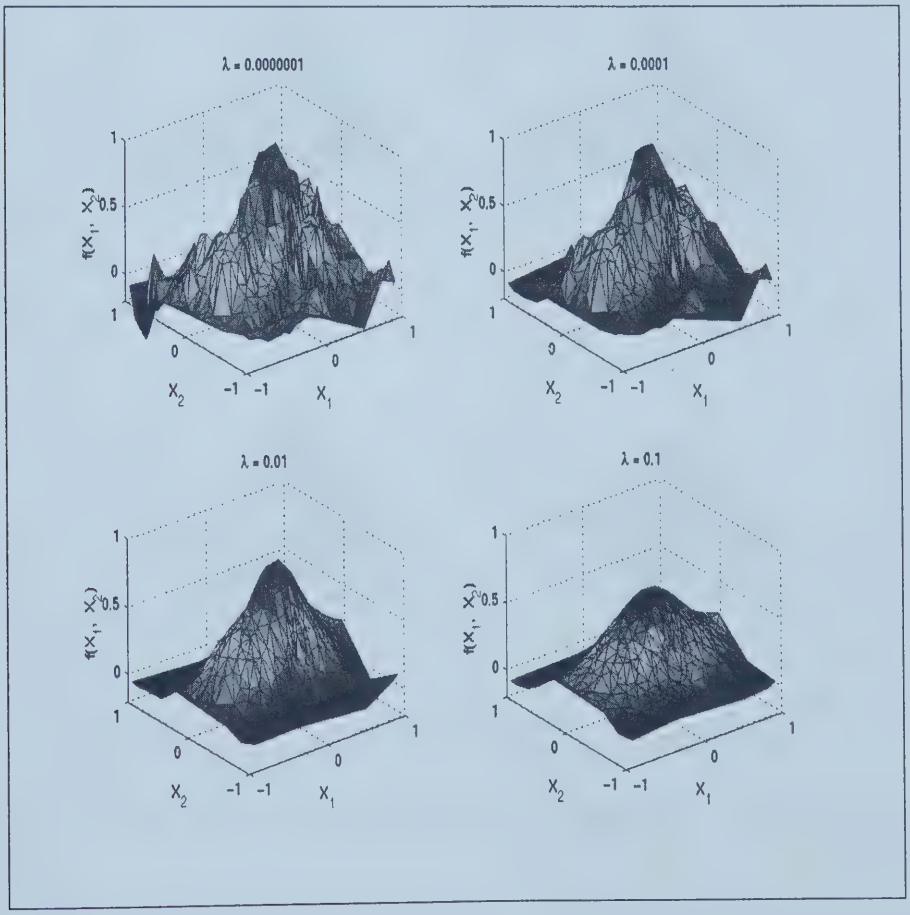

Figure 4.6: FE L-Spline's Bivariate Smooths of the Noisy Wendelberger Function.

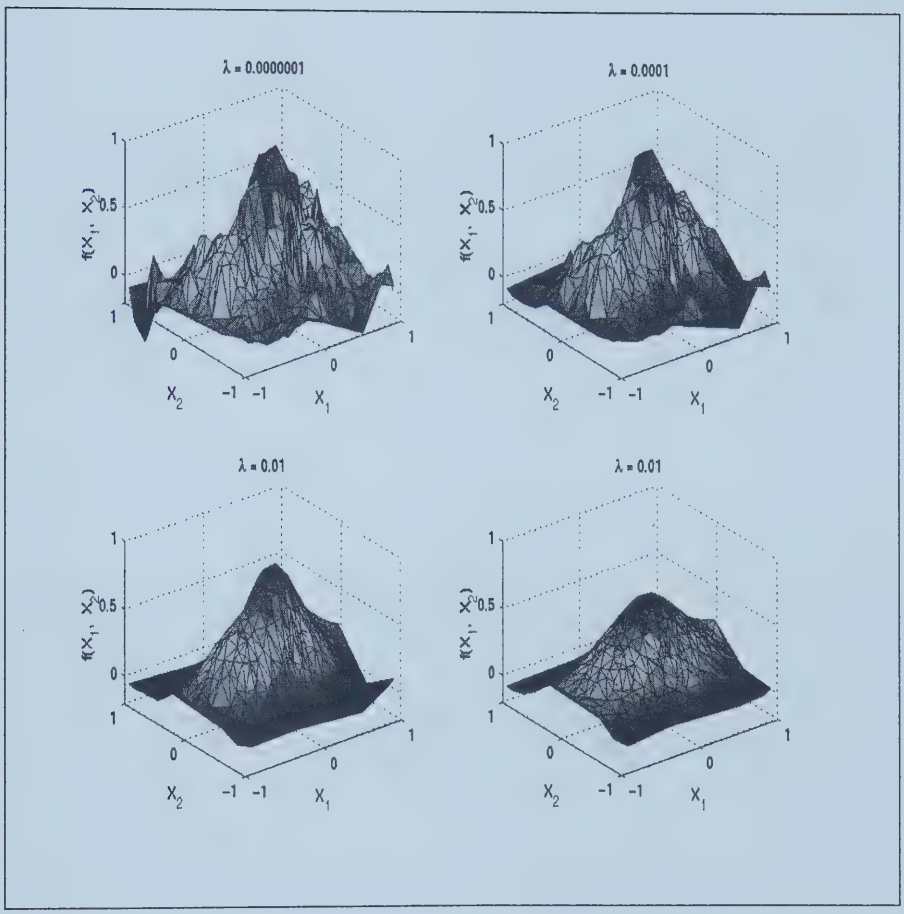

Figure 4.7: MFE L-Spline's Bivariate Smooths of the Noisy Wendelberger Function.

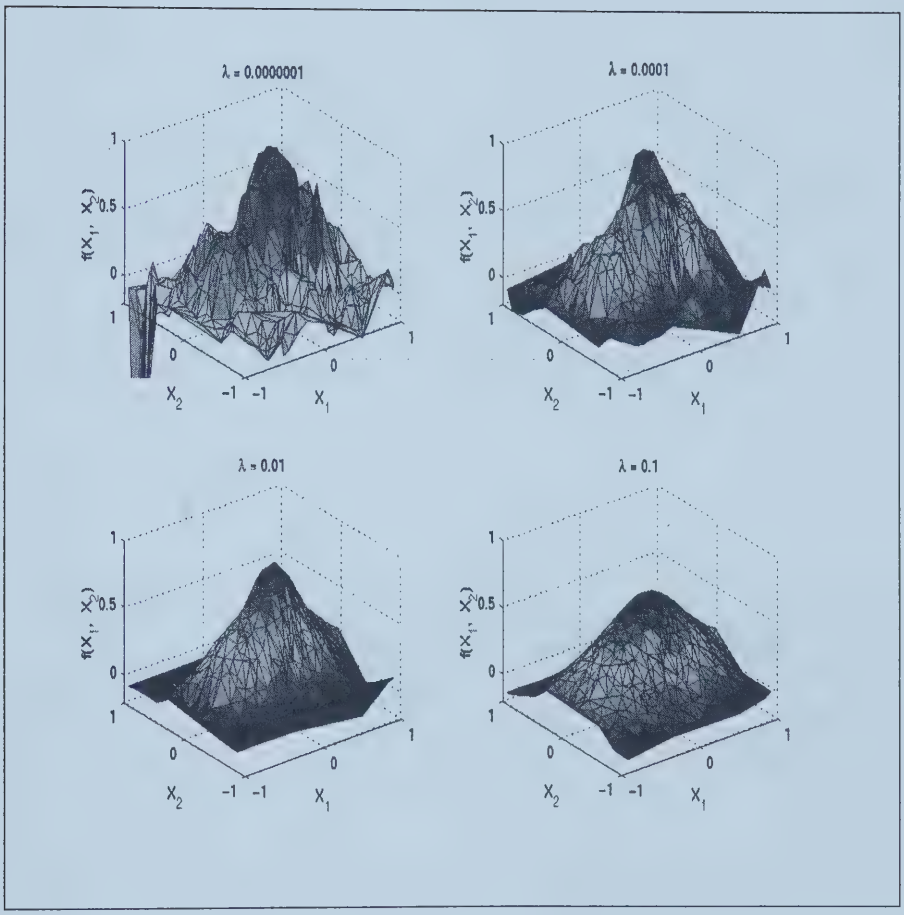

Figure 4.8: TPSFEM's Bivariate Smooths of the Noisy Wendelberger Function.

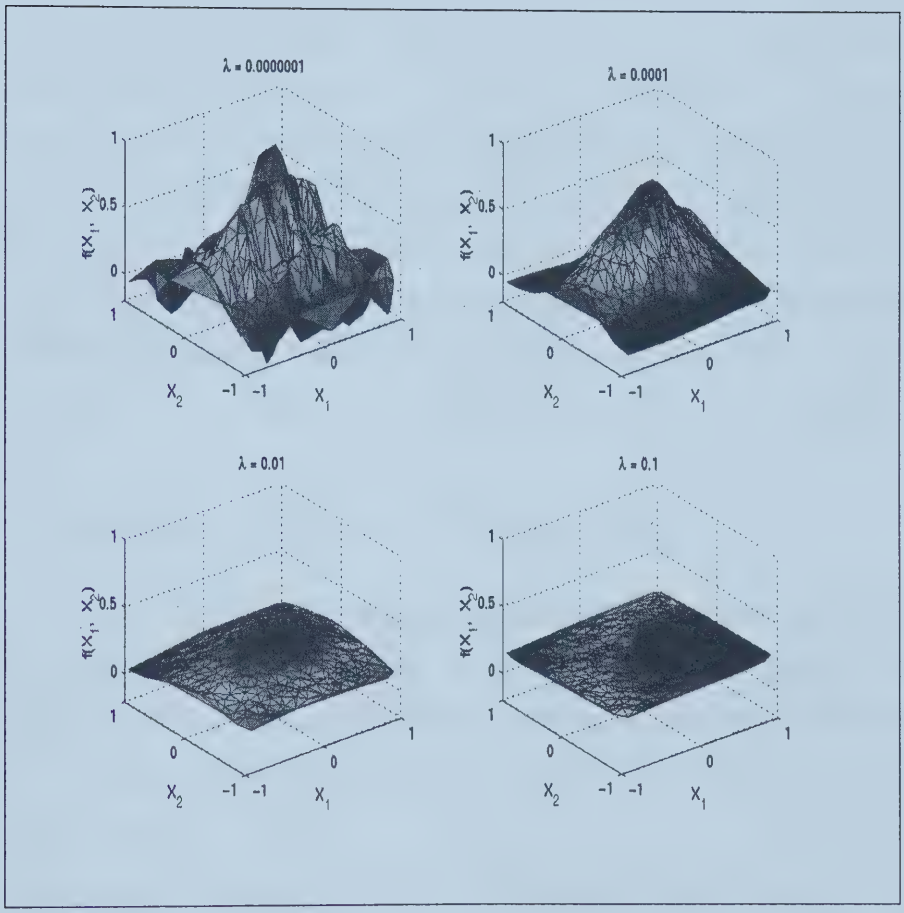

Figure 4.9: Thin-Plate Spline's Smooths of the Noisy Wendelberger Function.

4.2.3 Real Data Sets

Moreover, we apply these methods to real data sets which we refer to as the "ozone" and "Cobar mine" data. The ozone data consist of longitude and latitude components of 20 monitoring stations in Chicago, where the average daily ozone values during the period June-August 1987 are observed. In this data set, we consider longitude as the first regressor variable, latitude as the second regressor variable and the average daily ozone measure as response variable. The latter data set was collected by O'Connor and Leach (1979) from a mine in Cobar, Australia. The data set can be found in Green and Silverman (1994, p. 145). It contains 38 sampling locations (regressors) on the mine site, and the true "width" (response) of the ore bearing layer is measured at these locations. We also present the different regression fits of these data sets.

4.2.4 Generalized Smooths for Ozone Data

Stratospheric ozone is a relatively unstable substance which is created and destroyed in the stratosphere, kilometers above the earth's atmosphere. Although it occupies only a tiny portion of the stratosphere, stratospheric ozone is deemed crucial for life on earth. It acts as a shield to protect earth's surface from the sun's harmful ultra violet (UV) radiation.

Since the 1920's, stratospheric ozone has been measured from the ground. Scientists place instruments at specific locations to measure the amount of UV radiation that gets through the atmosphere; and then calculate the ozone concentration in the atmosphere above those locations. But now, NASA's Earth Observing System (EOS) measures ozone concentrations from space. Although data gathered are useful for learning more about ozone, these are still inadequate to describe the behavior of

global ozone concentrations. In addition, scientists are claiming that ozone is being depleted worldwide partly due to human activities. As a challenge, they are studying how to separate the amount of ozone depleted due to humans from those due to natural causes. If they will succeed then they can improve models for predicting ozone levels across the globe. However, accomplishing this will require an intense study on the interactions and factors affecting ozone's destruction and creation, which will still take many years.

While the decreasing stratospheric ozone may lead to skin cancer, ground-level ozone (air pollutants) causes damage to the human respiratory system and agricultural crops and trees (see Lippmann, 1989). There are two main kinds of air pollutants: *primary* and *secondary* pollutants. Primary pollutants are emitted from known sources while secondary pollutants are produced by chemical reactions between pollutants and other sources. The secondary pollutant O_3 is the most widely studied by many scientists. We will now attempt to extract local ozone characteristics by adding more regularly spaced monitoring sites in Chicago.

Figures 4.10-4.13 uncover the contours for the generalized FE-based smooths and the thin-plate spline's predictions for the ozone data. We see clearer structures of the data as the smoothing parameter λ goes to infinity. These figures also allow us to identify key sites in Chicago that might be of greater interest than the others.

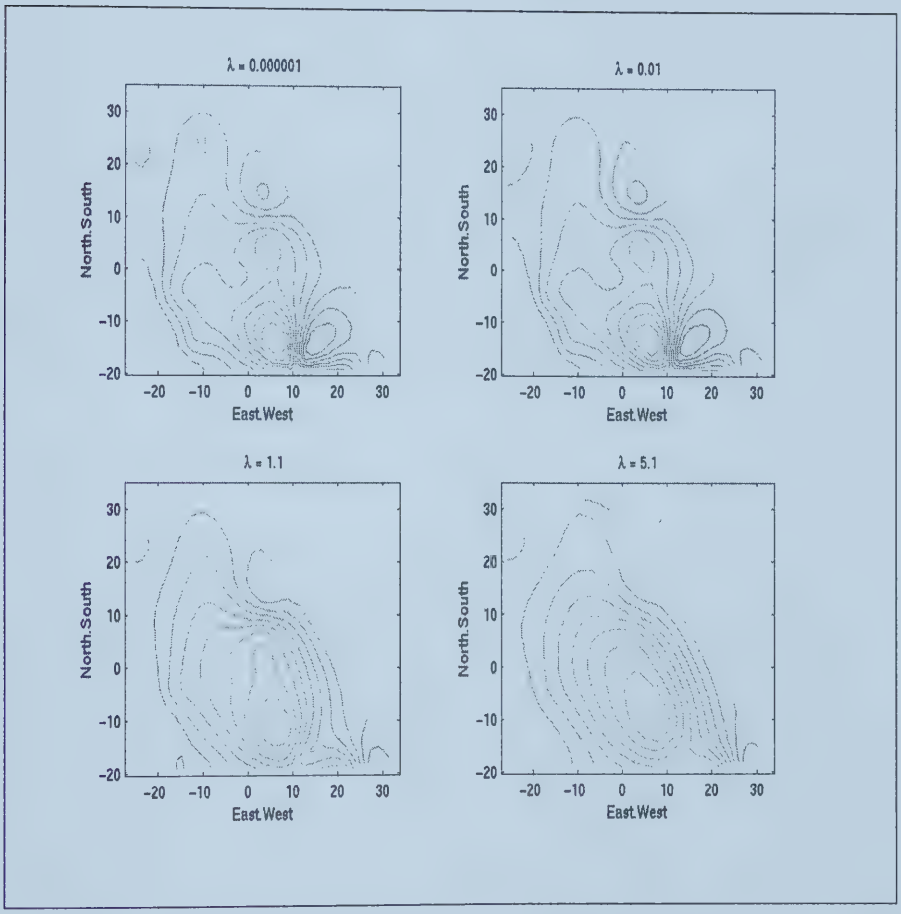

Figure 4.10: FE L-spline's Contour Plots of Ozone Data as $\lambda \to \infty$.

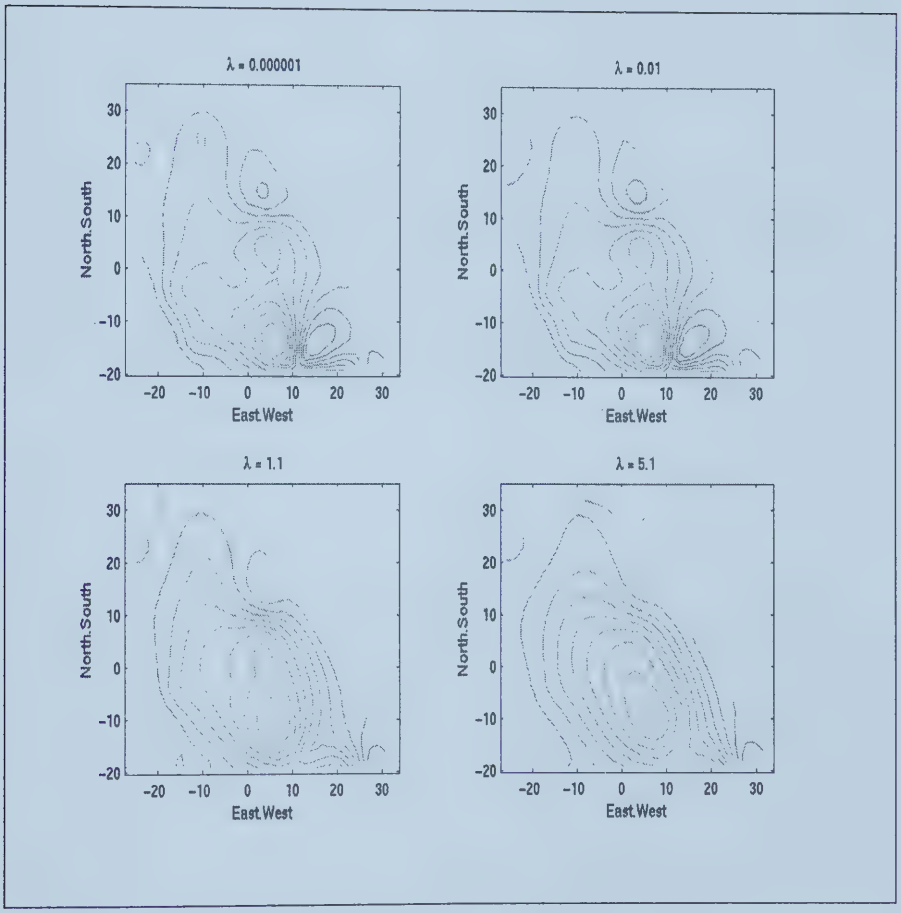

Figure 4.11: MFE L-spline's Contour Plots of Ozone Data as $\lambda \to \infty$.

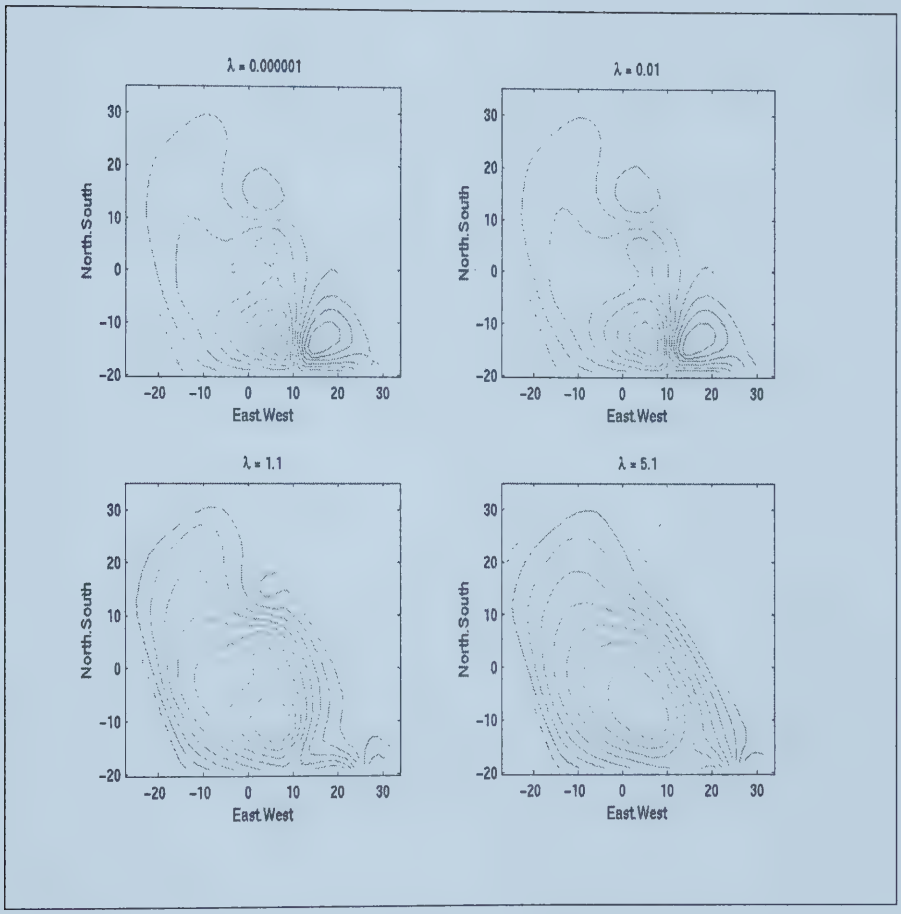

Figure 4.12: TPSFEM's Contour Plots of Ozone Data as $\lambda \to \infty$.

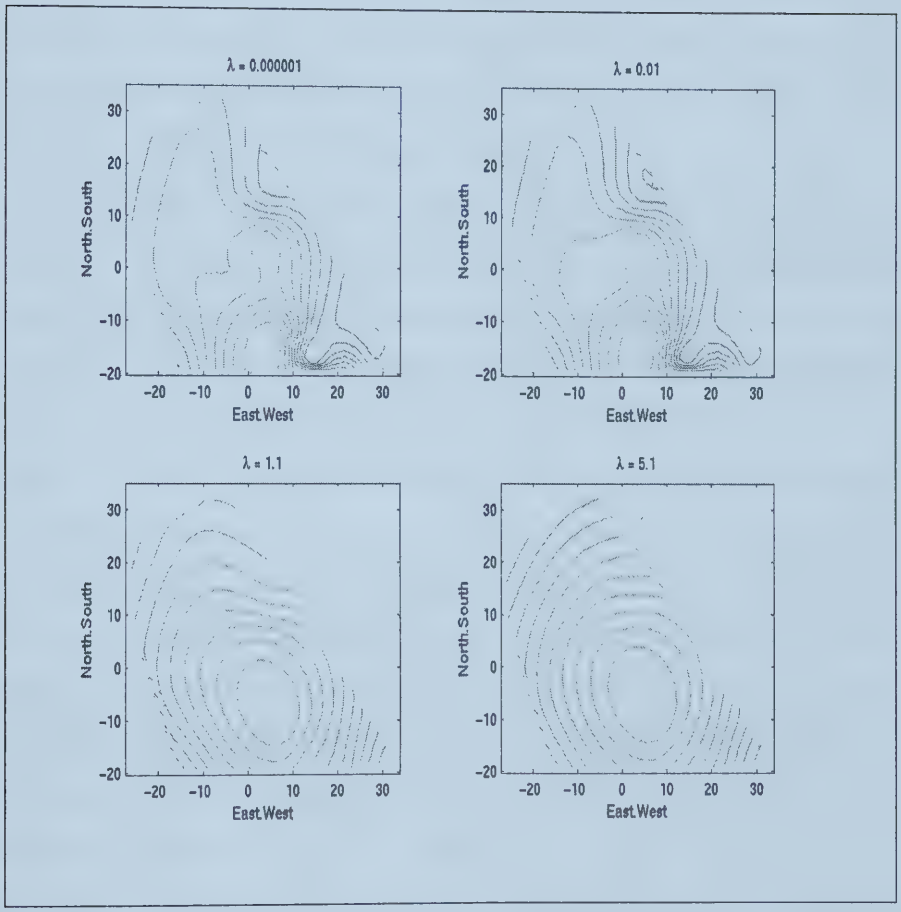

Figure 4.13: Thin-Plate Spline's Contour Plots of Ozone Data as $\lambda \to \infty$.

4.2.5 Generalized Smooths for Cobar Mine Data

Another important and interesting application is in the area of mining. In Cobar, NSW, Australia, substantial amounts of money have been spent on mine and near-mine explorations for acquiring advanced mining infrastructure. In the recent days, explorationists have been blamed for wasting valuable resources because of failures in discovering orebodies and other mineral deposits.

As a result, a local company named Peak Gold Mines Pty Limited (PGM) has spent millions to review and study key points that will lead to a successful near-mine exploration. In their study, they recommend some ways of how to optimize near-mine explorations, including maintaining consistent levels of exploration funding to prevent wastage; and ensuring that exploration programs are credible. We show some distributions of the true width of these ore bearing layers with new sampling sites added in Figures 4.14-4.17. These figures present the generalized bivariate FE-based smooths together with the standard thin-plate spline's prediction smooth. We can observe several attributes of the Cobar Mine data as the smoothing value λ increases. These figures also enable us to identify different onsets of smoothness among FE-based methods which will be helpful in mine and near-mine exploration. We see peaks, but nonetheless, the same trend of the generalized smooths can be observed as the smoothing parameter λ goes to infinity.

Meanwhile, Green and Silverman (1994) presented the contours of the smooths produced by standard thin-plate spline and finite window thin-plate spline for the Cobar mine data. They compared these two methods by looking at the contour of the difference between the corresponding interpolants with equal residual sum of squares. They stressed that the restriction to a finite domain affects the thin-plate

interpolant at the boundary. This suggests that thin-plate spline method somehow ignores smoothing values close to the boundary especially if there are a few data sites available. The contours similar to the ones produced by Stone (1988) in Green and Silverman (1994) are depicted in Figures 4.18-4.21. We also wish to replicate the graphs found in Green and Silverman (1994) but the algorithm for finite-window thin-plate spline was never published making it difficult to reproduce the exact plots.

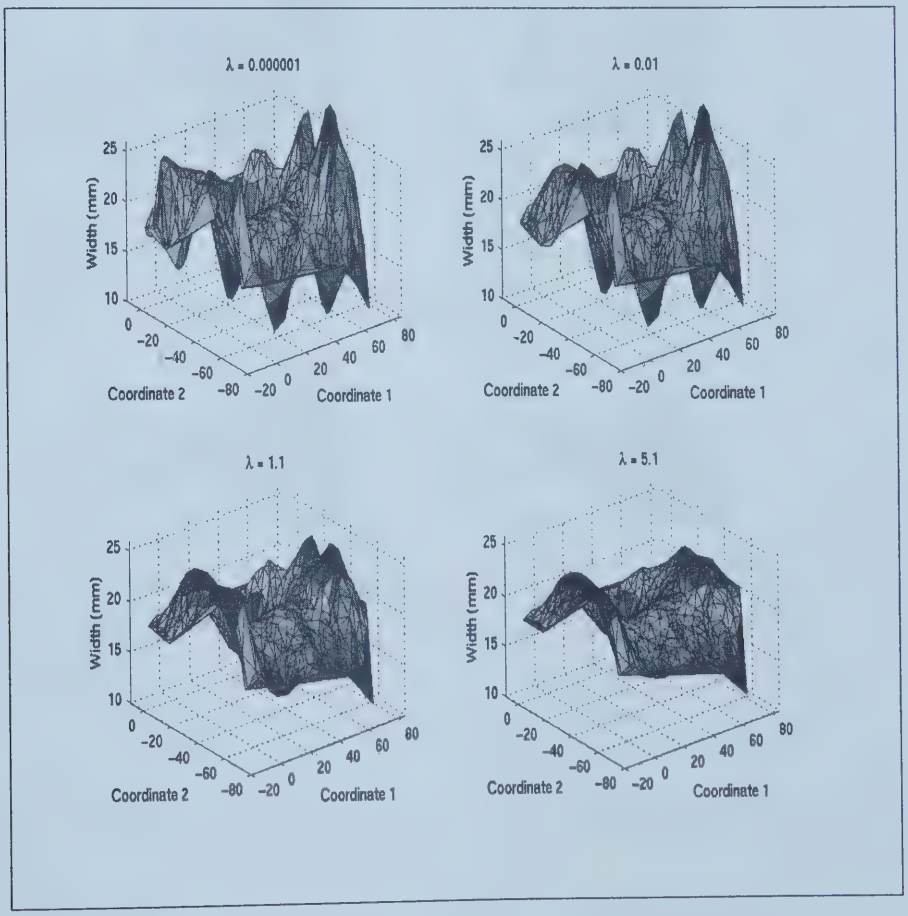

Figure 4.14: FE L-spline's Bivariate Smooths of the Cobar Mine Data as $\lambda \to \infty$.

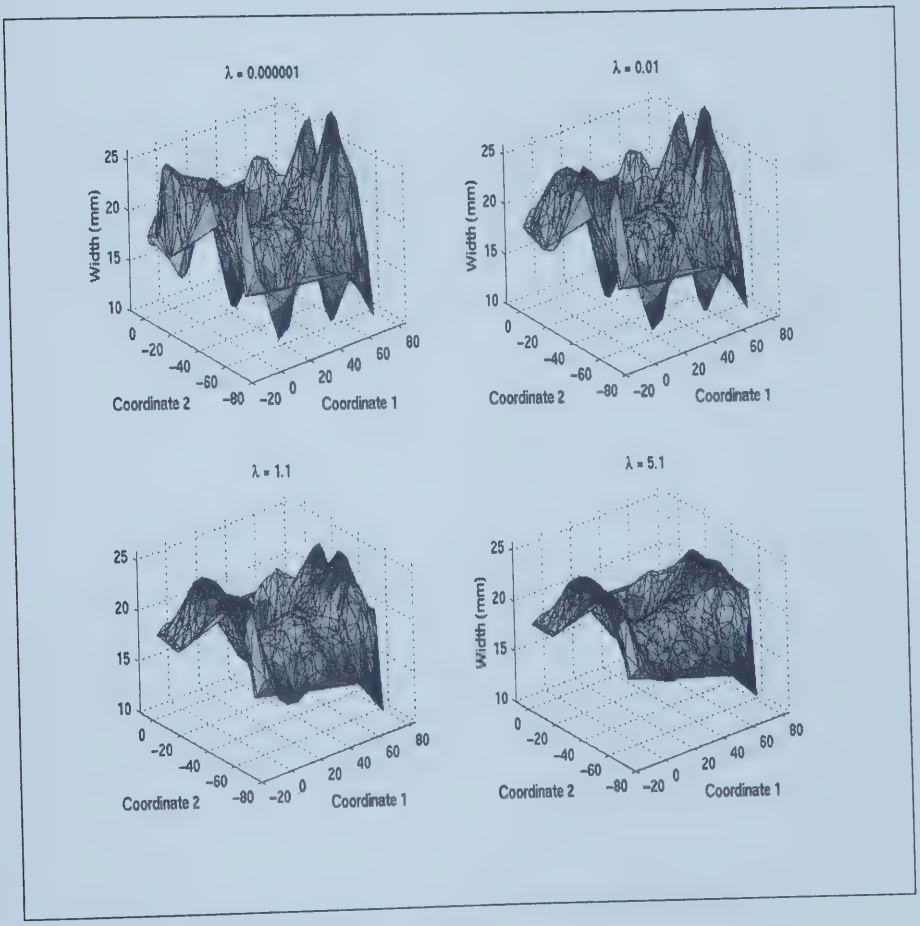

Figure 4.15: MFE L-spline's Bivariate Smooths of the Cobar Mine Data as $\lambda \to \infty$.

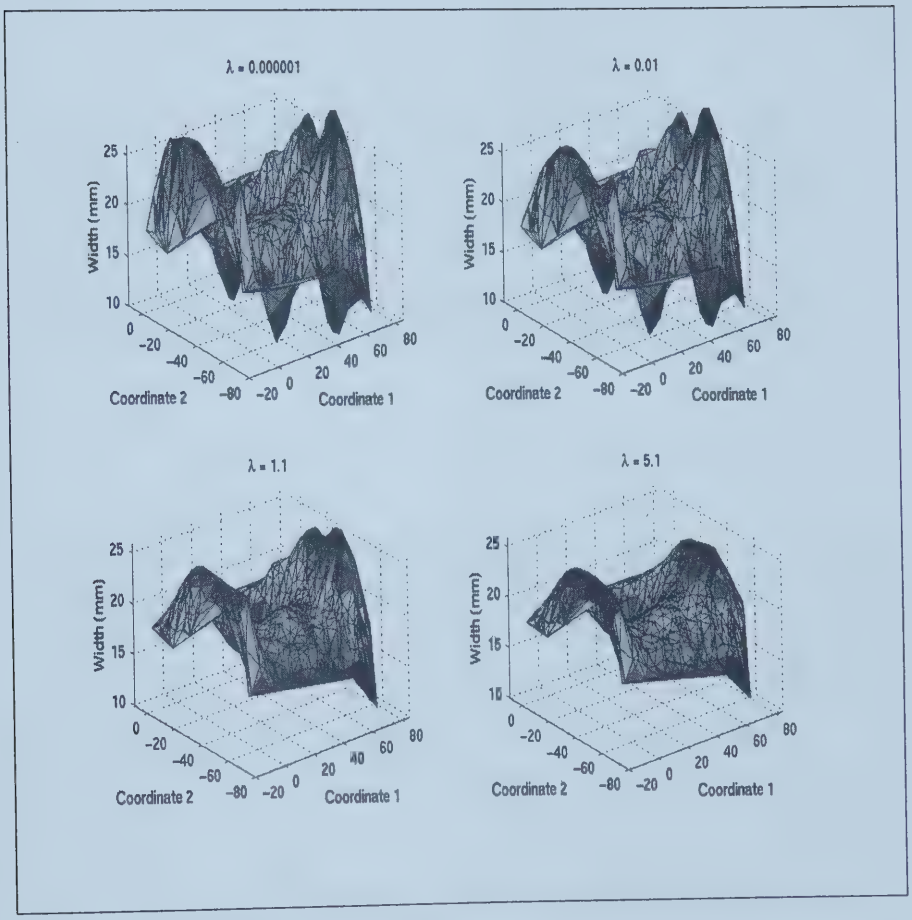

Figure 4.16: TPSFEM's Bivariate Smooths of the Cobar Mine Data as $\lambda \to \infty$.

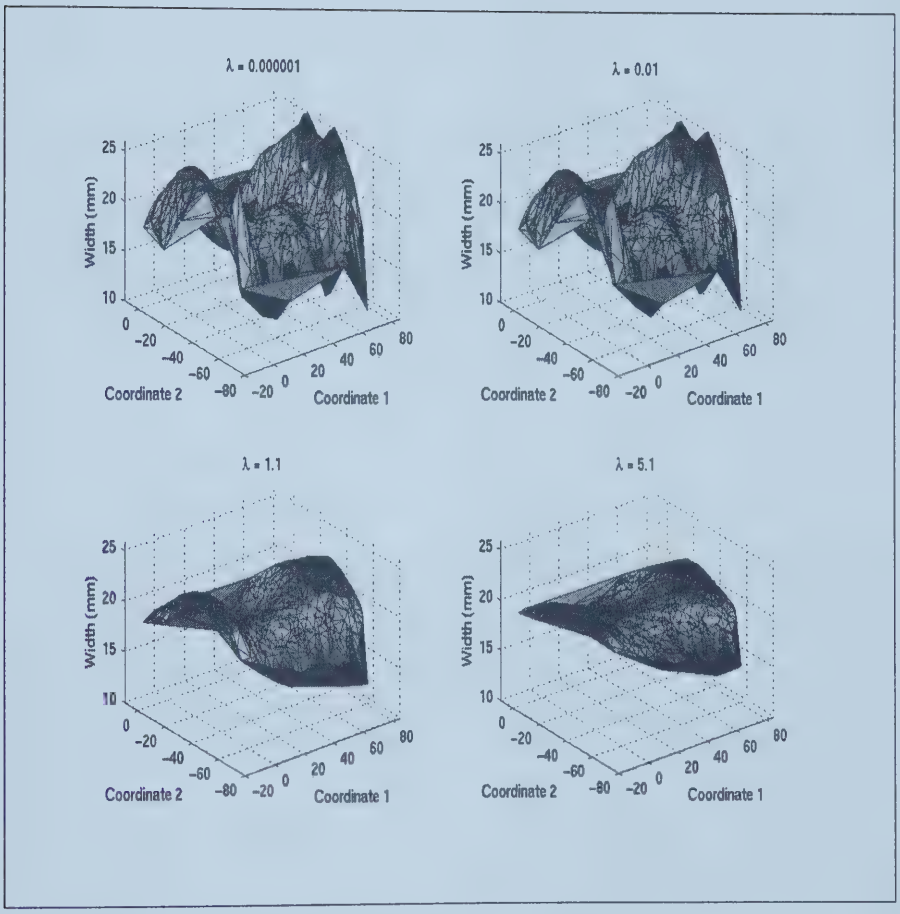

Figure 4.17: Thin-Plate Spline's Bivariate Fits of the Cobar Mine Data as $\lambda \to \infty$.

What follows are the corresponding contours. These plots reveal more vivid attributes of the data similar to the contours shown in Green and Silverman (1994).

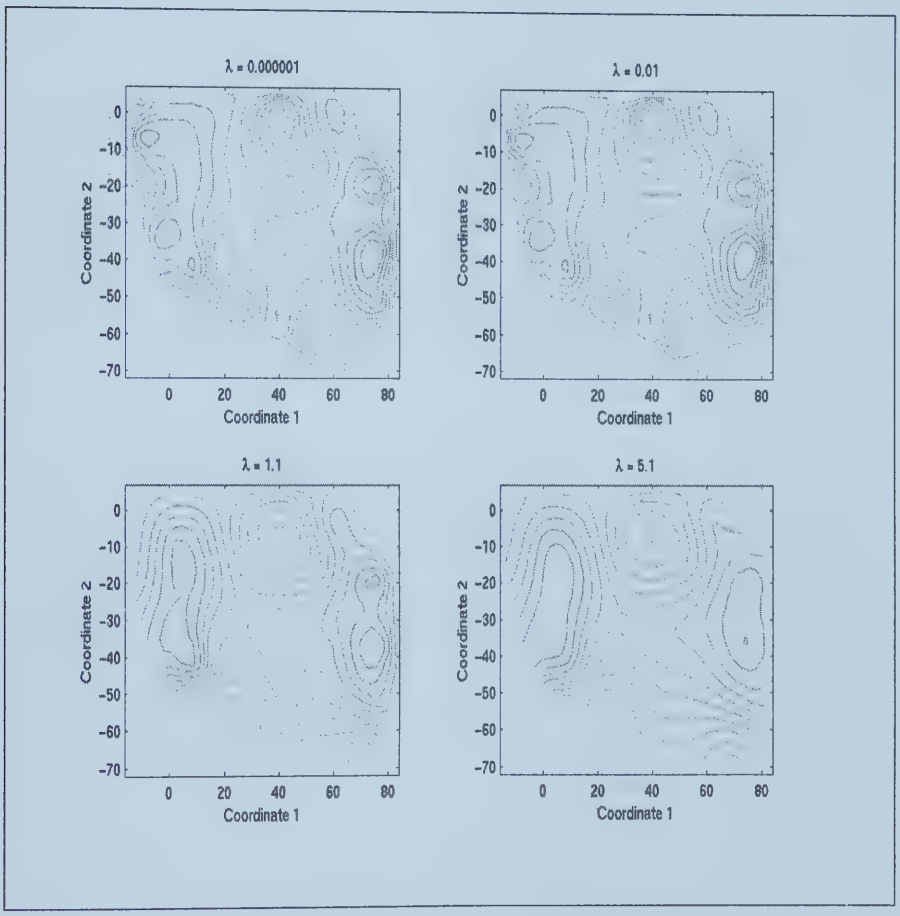

Figure 4.18: FE L-spline's Contour Plots of the Cobar Mine Data as $\lambda \to \infty$.

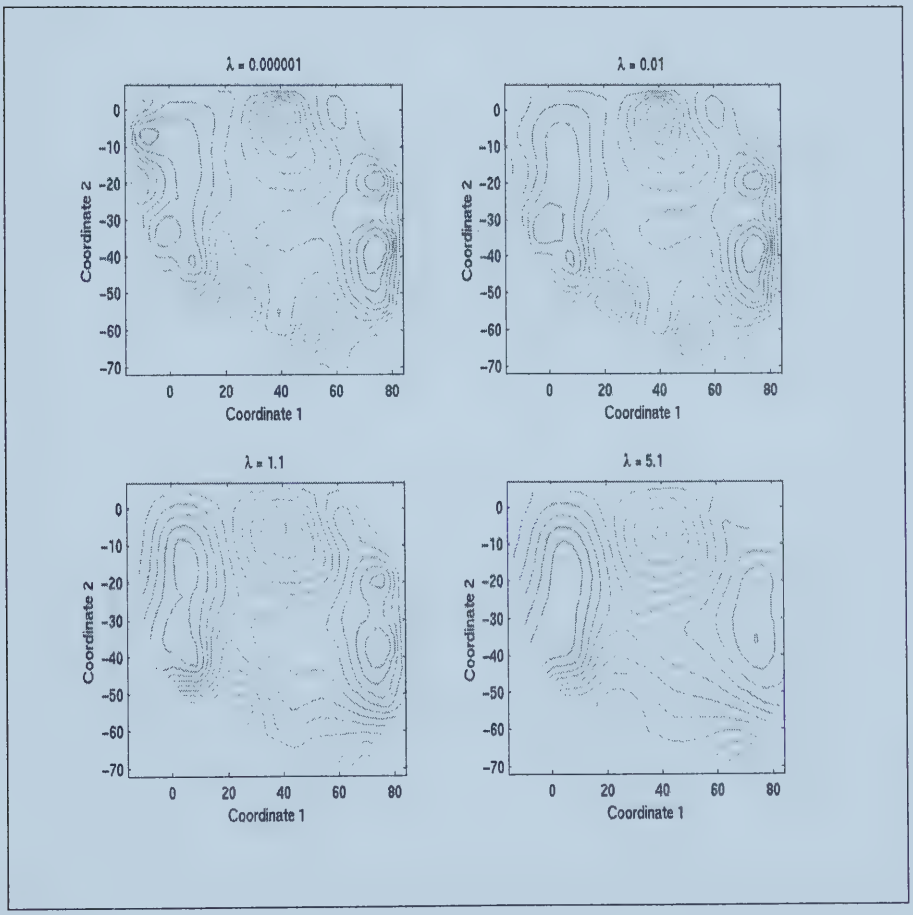

Figure 4.19: MFE L-spline's Contour Plots of the Cobar Mine Data as $\lambda \to \infty$.

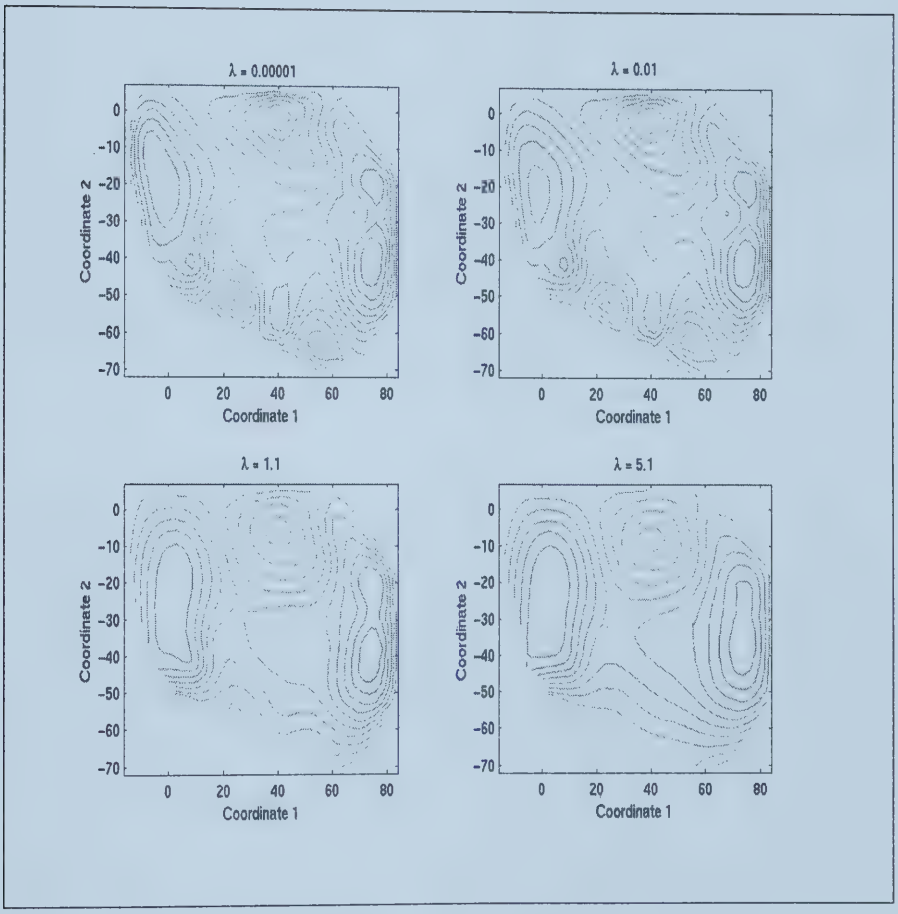

Figure 4.20: TPSFEM's Contour Plots of the Cobar Mine Data as $\lambda \to \infty$.

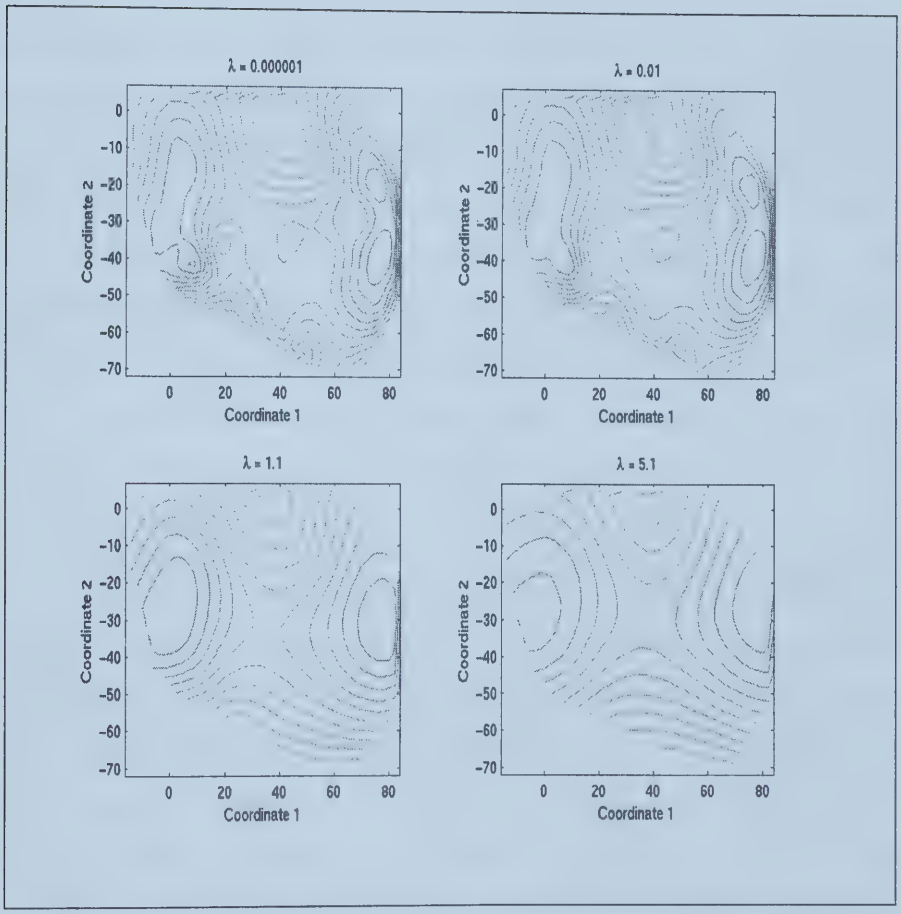

Figure 4.21: Thin-Plate Spline's Contour Plots of the Cobar Mine Data as $\lambda \to \infty$.

4.3 Interpretation of Smooths from Real Data Sets

4.3.1 Ozone Data

We know that smaller stratospheric ozone levels may lead to skin cancer while larger ground-level or surface ozone (O_3 pollutant) may damage human respiratory system. Since all the related literature (Nychka et al. (1996), Schimek (2000)) were silent about the kind of ozone the data is, we attempt to interpret the resulting smooths in two ways corresponding to both surface and stratospheric ozones. However, we state specifically the kind of ozone that we are referring to for clarity. Environmental scientists want to characterize ozone concentrations in Chicago from June to August of 1987. In so doing, they could be able to identify which areas in Chicago have high or low concentrations of ground-level or stratospheric ozone. They are also interested in determining the cyclic or periodic trend of ozone levels.

If the data is about ground-level or surface ozone then we offer the following interpretation: From Figures 4.10-4.13, we observe possible local maxima-minima which are represented by the holes and call them the "ozone holes". Note that for small smoothing parameter values ($\lambda = 0.000001$ and $\lambda = 0.01$, upper two panels), there are at least three holes which may be of interest. For instance, approximately at East.West = 5 and North.South = -15, ground-level ozone turns out to be minimum. This might suggest that people who have stayed in this particular area of Chicago may have smaller risks of developing respiratory problems than those who stayed outside the hole. However, at approximately East.West = 17 and North.South = -15, ground-level ozone is maximum. This may imply that individuals who stayed in this area may have increased symptoms of breathing disorders and may have worsened

the conditions of people who already have respiratory problems. In addition, we can generally observe that the maximum ozone holes are near the boundary of the convex hull formed by the data sites. This may show that the heart of Chicago area has still many locations which may be safe for individuals who have respiratory disorders. For larger smoothing parameter values ($\lambda = 1.1$ and $\lambda = 5.1$, lower two panels), it reveals only one hole which turns out to be a minimum. This may indicate that this site may be the only place in Chicago that has the least surface ozone level between June to August of 1987.

Similarly, if the data involves stratospheric ozone then we have the corresponding interpretation: From Figures 4.10-4.13, we see local maxima-minima which are represented by the "ozone holes". Note that for small smoothing parameter values ($\lambda = 0.000001$ and $\lambda = 0.01$, upper two panels), there are at least three holes which may be of interest. For instance, approximately at East.West = 5 and North.South = -15, ground-level ozone turns out to be minimum. This might suggest that people who have been significantly exposed to the sun's UV radiation in this particular area of Chicago may have larger risks of developing skin cancer than those who stayed outside the hole. However, at approximately East.West = 17 and North.South = -15, stratospheric ozone is maximum. This may imply that individuals who have been exposed to UV radiation in this area may have smaller risks of developing skin-related diseases especially skin cancer. Furthermore, we can also observe that the maximum ozone holes are near the boundary of the convex hull formed by the data sites. This may also show that the heart of Chicago area has many places where over-exposure to UV radiation may lead to serious skin problems. For larger smoothing parameter values ($\lambda = 1.1$ and $\lambda = 5.1$, lower two panels), it shows only one hole which turns out to be a minimum. This may indicate that this site is the only location in Chicago

that has the least stratospheric ozone level between June to August of 1987 and could likely be a skin-disease prone area.

Nychka et al. (1996) also offered a similar analysis for another ozone data in the United States. More specifically, they pointed out the relationship between a thin-plate spline and a kriging estimator in predicting ozone levels. Nychka further concluded that the ozone field is a nonstationary spatial process and that one should use some covariance structure that has different (marginal) variances but with isotropic (parametric) correlations. Several parametric forms of isotropic correlations are well discussed in the spatial statistics literature. More details about this study are available in Schimek (2000, chap. 13).

Furthermore, Nychka et al. (1996) expressed the idea of multiresolution analysis which allows decomposition (localization of basis functions) of the smooth at different resolutions. In smoothing, this is equivalent to comparing different smooths or estimates at decreasing values of λ. The decreasing λ values represent the multiresolution levels and the differences are called the "details" of the surfaces or smooths. This enables visualization of the change in the smooths as a function of the smoothing value λ. In this way, we can restrict λ to values which will give the detail surface that we want. More importantly, they pointed out that thin-plate and kriging estimators may likely miss local features of the data. This is so because these methods use global basis functions that are defined on the whole domain of interest. It is this basis characterization that inhibits these methods from possibly capturing local featrues of the data. This may be a major pitfall for these two methods.

We believe that detecting local characteristics in surface smoothing is feasible using finite element method because of the localized structures of its basis functions, enabling for localization of the smoothing parameter λ. We know that there are

observations which are unevenly distributed in the plane and that their smoothness or wiggliness also varies. For instance, it may be that one may observe many peaks in some parts or subregions of the domain and no peaks in some other subdomains, a scenario often happening in practice. Such a problem may be overcome by assigning or applying different amounts of smoothing to different subdomains, which can be done subsequently by localization of the smoothing parameter λ. This is feasible in finite element approach.

4.3.2 Cobar Mine Data

In a Cobar mine, administrators want to accurately predict the width of ore bearing layers at sites where there are no measurements. In this way, the company could minimize spending on workforce and exploration devices should there be an accurate technology for predicting the true width of these orebodies.

Should the sampling sites or locations be independent and regularly spaced, determining accurately the width of these ore bearing layers by smoothing would be very possible, with significantly reduced or minimal cost. Of course, it might not be that straightforward since one has to take into account the impacts of the major pitfalls of smoothing. One of them is the possibility of "*undersmoothing*" ($\lambda \to 0$). This is a serious problem because it will only indicate locations (probably erroneous and with high variance) that might have large (thick) or small deposits of ore bearing layers but it would not take into account the errors possibly incurred during the data gathering or exploration process. This works well if one has a prior knowledge that errors are negligible. Another possible pitfall is "*oversmoothing*" ($\lambda \to \infty$). This would also be undesirable since the focus now would be more on error reduction which

will more likely lead to biased estimates or excessive loss of information. In fact, if the smooth forms a flat surface (for very large λ) then this means that anywhere in the mine site has the same width of ore bearing layers, which may not generally make sense in mining. Thus, one has to achieve a balance between these two disadvantages so as not to jeopardize resources in exploring mine and near-mine explorations. This means selecting the most appropriate smoothing value λ in a most convincing way. In the succeeding section, we present the contours of the generalized smooths for optimal λ chosen by generalized cross-validation (GCV) method.

For this data set, we look at Figures 4.18-4.21 since they show clearer structures of the surfaces. These figures reveal several local maxima and minima. At $\lambda = 0.000001$ and $\lambda = 0.01$ (upper two panels), the coordinates (Coordinate 1, Coordinate 2) of the local maxima are (70, -40), (70, -20), (0, -35) and (-5, -5). This clearly suggests that there are at least four sites which have larger ore bearing deposits and are potential profit sources. On the other hand, there are sites also which may not be worth exploring. These are represented by the coordinates (20, -55), (50, -60) and (40, 0).

At $\lambda = 1.1$ and $\lambda = 5.1$ (lower two panels), the sites which have larger (local maxima for smaller values of λ) ore bearing layers remain to be maxima. This strongly suggests that these sites are likely to have larger ore bearing layers. However, the site with coordinates (40, -5) and its surrounding areas clearly reveals to be the only minimum as λ becomes large. So this warns ore mine explorers to avoid this site.

4.4 Parameter (λ) Selection by Generalized Cross-Validation (GCV) Method

In this section, we present the generalized smooths for the real data sets where the smoothing parameter λ is chosen by generalized cross-validation (GCV) method. Consider $\mathbf{P}(\lambda)$ as the $n \times n$ "smoothing" or projection matrix that maps the response vector $\mathbf{Y}$ into the predicted values, i.e.,

$$\mathbf{P}(\lambda)\mathbf{Y} = [\hat{f}((x)_1), \hat{f}(\mathbf{x}_2), \cdots, \hat{f}(\mathbf{x}_n)]^T. \tag{4.21}$$

The residuals are given by $(\mathbf{I}_n - \mathbf{P}(\lambda))\mathbf{Y}$ and $n - trace(\mathbf{P}(\lambda))$ is the degrees of freedom associated with residuals. The trace of $\mathbf{P}(\lambda)$ is a measure of the number of effective paramters in the spline representation.

An estimate of λ can be found by minimizing the generalized cross-validation (GCV) function

$$GCV(\lambda) = \frac{n\sum_{i=1}^{n}(\hat{f}(\mathbf{x}_i) - Y_i)^2}{(n - trace(\mathbf{P}(\lambda))^2} = \frac{n\|(\mathbf{I}_n - \mathbf{P}(\lambda)\mathbf{Y}\|^2}{(n - trace(\mathbf{P}(\lambda))^2}. \tag{4.22}$$

We minimize this function by doing grid search where the grids are $10^{-i}, i = 1, \cdots, 10$ and from 0.1 to some positive real constant c with an increment of 0.1. With a sample size of 800, we choose λ that has the minimum GCV score. Figures 4.22-4.23 present the contours of the generalized smooths for the real data sets indicating the minimum GCV scores.

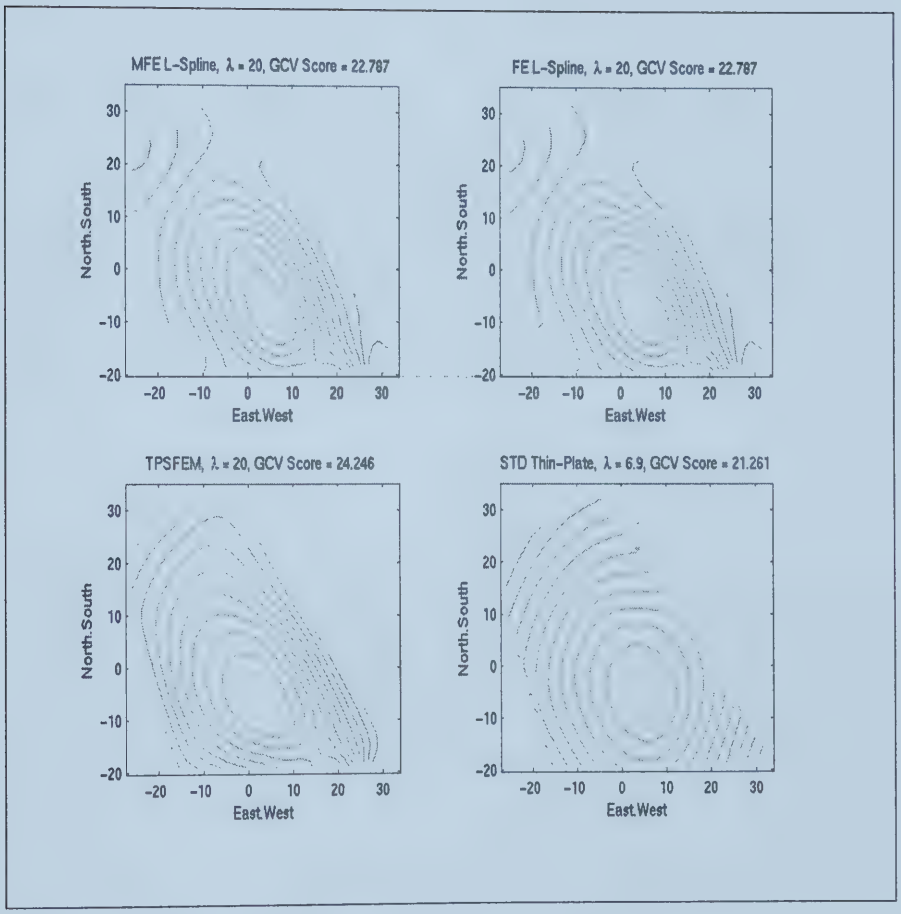

Figure 4.22: Contour Plots of Ozone Data where λ is Computed by GCV method.

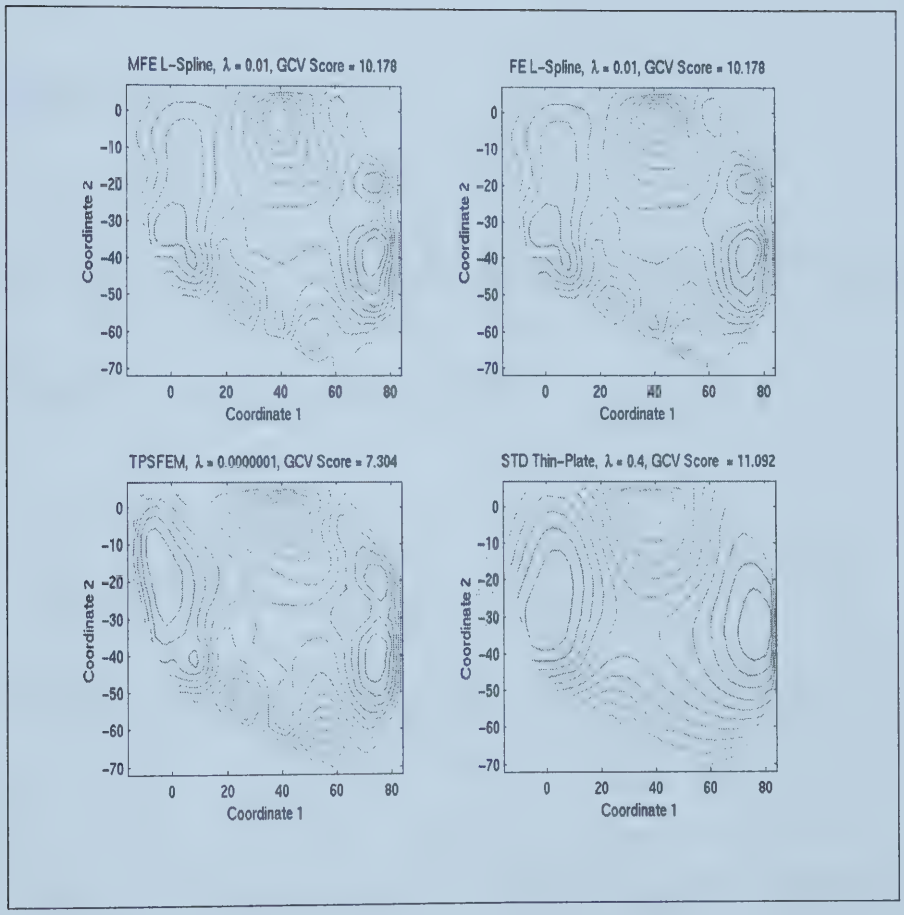

Figure 4.23: Contours of Cobar Mine Data where λ is Computed by GCV method.

Chapter 5

Summary, Conclusions and Future Research Directions

5.1 Summary

Firstly, we only considered fitting the following nonparametric regression model:

$$Y = f(X_1, X_2) + \epsilon, \tag{5.1}$$

where the errors are homogeneous coming from some unknown distribution and the dependence of the response variable Y depends on the regressor variables X_1 and X_2 is governed by some unknown joint distribution f. The relaxed assumption about the functional form of the relationship f made nonparametric regression more flexible in detecting structures of the data, identify candidate models, and allow better visualization of the data. Although the assumed nonparametric model is already flexible, there are still other forms of functional relationships which are also commonly used

in practice. For instance, there are relationships which may take the additive model, i.e.,

$$Y = f_1(X_1) + f_2(X_2) + \epsilon, \tag{5.2}$$

or a semi-parametric model, i.e.,

$$Y = \beta_1 X_1 + \beta_2 X_2 + g(X_3) + \epsilon. \tag{5.3}$$

The semi-parametric model (5.3) will be more efficient than the nonparametric model (5.1) if an auxiliary covariate X_3 is present and where the response variable is known to depend strongly. More details about the above models and their generalizations can be found in Hastie and Tibshirani (1990).

In the implementation process, we used finite element method (FEM) since it provides a more sensible foundation (assumes a bounded domain which is formed by the convex hull of the data sites) of solving real-life problems in contrast to the classical thin-plate spline method (assumes an infinite domain $\mathbb{R}^2$). More specifically, we implemented three finite element (FE) bivariate smoothing methods namely: finite element L-spline (FE L-spline), modified finite element L-spline (MFE L-spline) and finite element thin-plate spline method (TPSFEM). We emphasized that its solely the size of the sparse matrices involved in the corresponding linear systems that made the slight difference in the computational complexity or time among the three FE-based smoothing methods. Although comparison of the computational complexities/times of these FE-based methods was not formally done, each of these methods can be made faster than the others depending on how the codes are implemented and optimized.

While FE-based methods involved sparse matrices in the linear system, the classical thin-plate involved a full matrix. The sparsity property of the matrices in linear systems have better the computational efficiency of FE-based methods. This has

been the major advantage of FE approach over the classical thin-plate spline method especially for very large data sets. We also implemented the generalized versions of the three finite element methods for bivariate smoothing. We tested the FE-based methods by analyzing synthetic and real data sets; and presented smooths based on generalized cross-validation method. Lastly, we provided the Matlab code and Noweb documentation for interested individuals.

5.2 Conclusions

Results generally showed appealing and promising features of FEM in bivariate smoothing. As substantiated by the graphical results, FE-based methods are comparable with the standard thin-plate spline method. Moreover, the proposed modified FE L-spline (MFE L-spline) method provides an alternative procedure that is also comparable to the standard thin-plate spline method. The results obtained have been considerably substantial, albeit the study still calls for more investigations specifically in determining the exact limitations of these methods. Nevertheless, we conclude that these proposed FE-based methods can be used to smooth large data sets.

5.3 Possible Research Directions

Despite the intensive numerical investigations that have already taken place, there are still a number of things left undone that could be considered as possible extensions of the current exposition. These may include the following: compute the errors, bias and variance to assess accuracy; characterize and prove the convergence of the

generalized bivariate smooths to the true solution; provide rules on how to add or insert new design points; improve the algorithms implemented in Matlab to further its computational efficiency; attempt to use the Matlab code in smoothing and extracting valuable information from large databases; provide an automatic λ selection scheme; consider other interpolants, specifically higher degree polynomials; use other mesh generation methods aside from triangulation; extend the method to higher dimensions (e.g., having three or more regressors for some generalized additive models); provide a real data-driven FE smooth approximation algorithm that will not assume any specific form of the interpolant; and apply FE methods to solving other statistical problems.

Bibliography

[1] Becker E., Carey G., and Oden J. *Finite Elements : An Introduction*, volume 1. Prentice-Hall, New Jersey, 1981.

[2] Becker E., Carey G., and Oden J. *Finite Elements : A Second Course*, volume 2. Prentice-Hall, New Jersey, 1983.

[3] Becker E., Carey G., and Oden J. *Finite Elements : Mathematical Aspects*, volume 4. Prentice-Hall, New Jersey, 1983.

[4] Christen P., Altas I., Hegland M., and Roberts S. Parallelization of a finite element surface fitting algortihm for data mining. Technical report, Australian National University, May 2000.

[5] Christen P., Hegland M., Nielsen O., Roberts S., Strazdins P., and Altas I. Scalable parallel algorithms for surface fitting and data mining. Technical report, Australian National University, September 2000.

[6] de Boor C. *A Practical Guide to Splines.* Springer-Verlag , New York, 1978.

[7] Duchamp T. and Stuetzle W. Spline smoothing on surfaces. September 2001.

[8] Eubank R.L. *Spline Smoothing and Nonparametric Regression.* Dekker, New York, 1988.

[9] Flaherty J. *Finite Element Analysis.* Rensselaer Polytechnic Institute, New York, 2000.

[10] Green P.J. and Silverman B.W. *Nonparametric Regression and Generalized Linear Models, A Roughness Penalty Approach.* Chapman and Hall, London, 1994.

[11] Härdle W. *Applied Nonparamateric Regression.* Cambridge University Press, Boston, 1991.

[12] Hastie T.J. and Tibshirani R.J. *Generalized Additive Models.* Chapman and Hall, London, 1990.

[13] Heckman N. The theory and application of penalized least squares methods or reproducing kernel hilbert spaces made easy. Technical report, University of British Columbia, August 1997.

[14] Heckman N. and Ramsay J. Penalized regression with model-based penalties. Technical report, University of British Columbia, 2000.

[15] Hegland M., Roberts S., and Altas I. Finite element thin plate splines for surface fitting. *in Computational Techniques and Applications: CTAC97,Singapore, World Scientific*, pages 289–296, 1997.

[16] Hegland M., Roberts S., and Altas I. Finite element thin plate splines for surface fitting. *in Mathematical Methods for Curves and Surfaces, II (Lillehammer, 1997), Vanderbilt Univer. Press, Nashville, TN*, pages 245–252, 1998.

[17] Johnson C. *Numerical Solution of Partial Differential Equations by the Finite Element Method.* Press Syndicate of the University of Cambridge,Cambridge, 1987.

[18] Koenker R. and Mizera I. Penalized triograms: total variation regulation for bivariate smoothing. preprint. 2001.

[19] Lippman M. Health effects of ozone : A critical review. *J. of the Air Waste Management Association*, 39:672–695, 1989.

[20] Nychka C., Bailey B., Ellner S., Haaland P., and O'Connell M. FUNFITS data analysis and statistical tools for estimating functions. *North Carolina Institute of Statistics Mimeoseries*, No. 2289, 1996.

[21] Ramsay T.O. *A Bivariable Finite Element Smoothing Spline Applied to Image Registration.* PhD thesis, Queen's University, 1999.

[22] Ramsay J.O. and Silverman B.W. *Functional Data Analysis.* Springer-Verlag, New York, 1997.

[23] Ramsay J.O. and Heckman N. Some theory for L-spline smoothing. Technical report, University of British Columbia, July 1996.

[24] Reddy J.N. *Applied Functional Analysis and Variational Methods in Engineeering.* McGraw-Hill Book Company, New York, 1986.

[25] Roberts S., Hegland M., and Altas I. Approximation of a thin plate spline smoother using a continuous piecewise polynomial functions. *SIAM J. Numer. Anal.*, 41:208–234, 2003.

[26] Schimek M. *Smoothing and Regression : Approaches, Computation and Application.* John Wiley Sons, USA, 2000.

[27] Shumaker L. *Spline Functions : Basic Theory.* John Wiley and Sons Inc., New York, 1981.

[28] Sibson R. and Stone G. Computation of thin-plate splines. *SIAM Journal of Statistical and Scientifc Computing*, 12:1304–1313, 1991.

[29] Silverman B.W. *Density Estimation for Statistics and Data Analysis.* Chapman and Hall, New York, 1986.

[30] Stone G. *Bivariable Splines.* PhD thesis, University of Bath, 1988.

[31] Strang G. and Fix G. *An Analysis of Finite Element Method.* Prentice-Hall Inc., 1973.

[32] Tarter M.E. and Lock M.D. *Model-Free Curve Estimation.* Chapman and Hall, New York, 1993.

[33] Wahba G. *Spline Models for Observational Data.* Philadelphia, Pa: SIAM, 1990.

[34] Wahba G. Splines in nonparametric regression. Technical report, University of Wisconsin-Madison, September 2000.

[35] Zienkiewicz O.C. and Taylor R.L. *The Finite Element Method*, volume 1. The Basis. Butterworth-Heinemann, 5th edition, 2000.

Appendix

NOWEB-Processed Documentation

Objective

This article aims to describe the computation of bivariate smooths using finite element L-spline (FE L-spline), finite element thin-plate spline (TPSFEM) and the modified FE L-spline (MFE L-spline) procedures. This is tested in Matlab version 6.1.0.450 Release 12.1 for Unix. For details, please see the corresponding manuscript.

The SubAlgorithms

The following function `TriData.m` stores and loads the data, its dimensions and other information neccesary as inputs for the succeeding calculations.

⟨*TriData.m*⟩≡

```
function LoadData = TriData( Y, X1, X2)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% INPUTS :
%           Y   = Column of Values of the Response Variable
%           X1  = regressor variable 1
%           X2  = regressor variable 2
% OUTPUTS :
%           tri     = nodal indices obtained after triangulation
%           nelem   = number of elements or triangles
%           nnodes  = number of original data points
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
tri =delaunay(X1,X2);
[nbounds, junk]= size(boundary);
[nelem, junk]=size(tri);
[nnodes,junk]=size(X1);
LoadData = { tri Y X1 X2 nelem nnodes };
```

After inputting the data set, nodal components or indices for each element or triangle will be automatically loaded and re-ordered in a counter clockwise manner by `CcWise.m`. This is necessary for computing the coefficients of the linear interpolants later.

⟨*CcWise.m*⟩≡

```
function flip = CcWise(LoadData)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% This rearranges the triangular(local) nodal points into a
% counter clockwise order
% INPUT :
%     LoadData = object produced by TriData, which includes the
%                components of the triangles/elements
% OUTPUT :
%    tri2 = tringular components with counter clockwise ordering
%            for each triangle
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
tri = LoadData{1};
X1 = LoadData{3};
X2 = LoadData{4};
nelem = LoadData{5};
tri2 = sparse(nelem,3);
for i=1:nelem
     if ((det([ 1 X1(tri(i,1)) X2(tri(i,1));...
                1 X1(tri(i,2)) X2(tri(i,2));...
                1  X1(tri(i,3))  X2(tri(i,3)) ])) < 0)
        tri2(i,:) = fliplr(tri(i,:));
     else
        tri2(i,:) = tri(i,:);
     end;
```

```
end;
flip = tri2;
```

The area and the coefficients of the linear functions for each triangle or element are then computed by `AreaCoef.m`. The area is utilized here for computing the integrals in the assembly of matrices.

⟨*AreaCoef.m*⟩≡

```
function coef = AreaCoef(flip, LoadData)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%This computes the area of each triangular element and
% the coefficients of the linear interpolant
% INPUT :    flip   = matrix produced by CcWise.m
%           LoadData= object constructed by TriData.m
%OUTPUTS:   area   = area of each tringular element
%            B1     = "a" or the constant part of the
%                     linear interpolant a + b(X1) + c(X2)
%            B2     = "b" part
%            B3     = "c" part
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
tri = LoadData{1};
X1 = LoadData{3};
X2 = LoadData{4};
nelem = LoadData{5};
```

```
nnodes =LoadData{6};
tri2 =flip;
%area of the each triangle
area = sparse(nelem,1);
for i=1:nelem
   area(i,1) = abs((0.5)*det([ 1 X1(tri2(i,1)) X2(tri2(i,1));...
                               1 X1(tri2(i,2)) X2(tri2(i,2));...
                               1 X1(tri2(i,3)) X2(tri2(i,3)) ]));
end;
tri2 = flip;
%computing the coefficients
B11=sparse(nnodes,nelem);
B22=sparse(nnodes,nelem);
B33=sparse(nnodes,nelem);
for i=1:nnodes
       for j=1:nelem
          dummy =sparse(3,1);
          C =sparse(1, 3);
       if (tri2(j,1)==i)
       dummy = (inv([ 1 X1(tri2(j,1)) X2(tri2(j,1));...
                      1 X1(tri2(j,2)) X2(tri2(j,2));...
                 1 X1(tri2(j,3)) X2(tri2(j,3))])*[1; 0 ; 0]);
       B11(i,j) = dummy(1,1);
       B22(i,j) = dummy(2,1);
```

```
        B33(i,j) = dummy(3,1);
        elseif  (tri2(j,2)==i)
        C = tri2(j,[ 2 1 3 ]);
        dummy = (inv([ 1 X1(C(1,1)) X2(C(1,1));...
                       1 X1(C(1,2)) X2(C(1,2));...
                       1 X1(C(1,3)) X2(C(1,3))])*[1; 0 ; 0]);
        B11(i,j) = dummy(1,1);
        B22(i,j) = dummy(2,1);
        B33(i,j) = dummy(3,1);
        elseif (tri2(j,3)==i)
        C =fliplr(tri2(j,:));
        dummy = (inv([ 1 X1(C(1,1)) X2(C(1,1));...
                       1 X1(C(1,2)) X2(C(1,2));...
                      1 X1(C(1,3)) X2(C(1,3))])*[1; 0 ; 0]);
        B11(i,j) = dummy(1,1);
        B22(i,j) = dummy(2,1);
        B33(i,j) = dummy(3,1);
        elseif any(tri2(j,:)~= i)
        B11(i,j) = 0;
        B22(i,j) = 0;
        B33(i,j) = 0;
        end;
    end;
end;
```

```
B1=B11';%constant
B2=B22';%coef for X1
B3=B33';%coef for X2
coef = { area B1 B2 B3};
```

The information extracted from `AreaCoef.m` and `TriData.m` are then used to calculate the matrix $\mathbf{Q} = \int(\psi_{x_1}\psi_{x_1}^T + \psi_{x_2}\psi_{x_2}^T)$.

⟨*MinusLap.m*⟩≡

```
function Q = MinusLap(coef, LoadData)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% INPUTS:
%            coef         = object produced by AreaCoef.m
%            LoadData     = object processed by TriData.m
% OUTPUT    : Q or L   = the matrix approximation to the minus
%                        Laplacian operator or the integral of the
%                        product between the gradient vector of
%                        the basis function and its transpose
%                        under L^2.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
nnodes = LoadData{6};
area = coef{1};
B1 = coef{2};
B2 = coef{3};
```

```
B3 = coef{4};
L2= sparse(nnodes);
L = sparse(nnodes);
for i=1:nnodes
  for j=i:nnodes
   if (j==i)
   L2(i,j)=(sum(( B2(:,i).*B2(:,j) + B3(:,i).*B3(:,j)).*area))/2;
   else
   L2(i,j) = sum(( B2(:,i).*B2(:,j) + B3(:,i).*B3(:,j)).*area);
   end;
  end;
end;
L= L2 + L2';
Q = L;
```

The construction and approximation of $\mathbf{C_1}$ and $\mathbf{C_2}$ are done by `DifOp.m` where $\mathbf{C_1} = \int_\Omega \psi_{x_1} \psi^T$ and $\mathbf{C_2} = \int_\Omega \psi_{x_2} \psi^T$. The subalgorithm also computes the averages of the nodal component values which are used in the construction of another matrix $\mathbf{P}$. It is also emphasized that the approximation of these integrals is done in a different way. That is, we make use of existing results and formulae in computing these integrals.

⟨ *DifOp.m* ⟩≡

```
function C1C2 = DifOp(flip, coef, LoadData)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
% INPUTS :
%           coef    = object produced by AreaCoef.m
%           flip    = object produced by CcWise.m
%           LoadData = object produced by TriData.m
% OUTPUTS:
%        C1      = matrix approximation to the partial differential
%                     operator w.r.t X1
%        C2      = matrix approximation to the partial differential
%                   operator w.r.t X2
%        aver_x1 = averages of all X1 components by element/triangle
%        aver_x2 = averages of all X2 components by element/triangle
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%C1 & C2 matrices assembly
tri2 = flip;
area = coef{1};
B1 = coef{2};
B2 = coef{3};
B3 = coef{4};
nelem = LoadData{5};
nnodes = LoadData{6};
X1 = LoadData{3};
X2 = LoadData{4};
% averages for each triangle across all elements
aver_x1= sparse(nelem,1);
```

```
aver_x2= sparse(nelem,1);
for i = 1:nelem
    aver_x1(i,1) = mean([ X1(tri2(i,1))  X1(tri2(i,2)) X1(tri2(i,3))]);
    aver_x2(i,1) = mean([ X2(tri2(i,1))  X2(tri2(i,2)) X2(tri2(i,3))]);
end;
C1 = sparse(nnodes);
C2 = sparse(nnodes);
for i =1:nnodes
  for j=1:nnodes
    C1(i,j) = sum( (B2(:,i).*B1(:,j)  + B2(:,i).*B2(:,j).*(aver_x1)+...
     B2(:,i).*B3(:,j).*(aver_x2)).*(area));
    C2(i,j) = sum( (B3(:,i).*B1(:,j)  + B3(:,i).*B2(:,j).*(aver_x1)+...
     B3(:,i).*B3(:,j).*(aver_x2)).*(area));
  end;
end;
C1C2 = {C1 C2 aver_x1 aver_x2};
```

What follows is the approximation of the matrix $\mathbf{P} = \int(\psi\psi^T)$ where the outputs of `DifOp.m`, `CcWise.m`, `AreaCoef.m` and `TriData.m` are used in the construction.

⟨*Pobject.m*⟩≡

```
function P = Pobject( flip, C1C2, coef, LoadData)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% INPUTS :
%           flip    = object processed by CcWise.m
```

```
%           LoadData= object procssed by TriData.m
%           C1C2    = object produced by DifOp.m
%           coef    = object produced by AreaCoef.m
% OUTPUT :  P = matrix approximation to the dot product between
%                 the vector of basis functions and its transpose
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
tri2=flip;
aver_x1 = C1C2{3};
aver_x2 = C1C2{4};
area = coef{1};
B1 = coef{2};
B2 = coef{3};
B3 = coef{4};
X1 = LoadData{3};
X2 = LoadData{4};
nelem = LoadData{5};
nnodes = LoadData{6};
%differences
difx11= sparse(nelem,1);
difx12= sparse(nelem,1);
difx13= sparse(nelem,1);
difx21= sparse(nelem,1);
difx22= sparse(nelem,1);
difx23= sparse(nelem,1);
```

```
for i = 1:nelem
        difx11(i,1) = X1(tri2(i,1))-aver_x1(i,1);
        difx12(i,1) = X1(tri2(i,2))-aver_x1(i,1);
        difx13(i,1) = X1(tri2(i,3))-aver_x1(i,1);
        difx21(i,1) = X2(tri2(i,1))-aver_x2(i,1);
        difx22(i,1) = X2(tri2(i,2))-aver_x2(i,1);
        difx23(i,1) = X2(tri2(i,3))-aver_x2(i,1);
end;
%P Matrix Assembly
Ko = sparse(nnodes);
K = sparse(nnodes);
for i =1:nnodes
    for j=i:nnodes
     dummy1 =sparse(nelem,1);
     dummy2 =sparse(nelem,1);
     dummy3 =sparse(nelem,1);
     dummy4 =sparse(nelem,1);
     dummy5 =sparse(nelem,1);
     dummy6 =sparse(nelem,1);
     dummy7 =sparse(nelem,1);
   if (j==i)
  dummy1=(B1(:,i).*B1(:,j)+(B1(:,i).*B2(:,j)+...
        B1(:,j).*B2(:,i)).*(aver_x1));
  dummy2=(B1(:,i).*B3(:,j) + B1(:,j).*B3(:,i)).*(aver_x2);
```

```
dummy3=(B2(:,i).*B3(:,j) + B2(:,j).*B3(:,i)).*((1/12)*...
       ones(nelem,1)).*(diffx11.*diffx21+...
        difx12.*difx22 +difx13.*dffx23);
dummy4=(B2(:,i).*B3(:,j) + B2(:,j).*B3(:,i)).*(aver_x1.*aver_x2);
dummy5=(B2(:,i).*B2(:,j)).*( difx11.^2 +difx12.^2 +difx13.^2).*...
     ((1/12)*ones(nelem,1));
dummy6=(B2(:,i).*B2(:,j).*(aver_x1.^2))+...
     (B3(:,i).*B3(:,j).*(aver_x2.^2));
dummy7=(B3(:,i).*B3(:,j)).*( difx21.^2 +difx22.^2 +difx23.^2).*...
     ((1/12)*ones(nelem,1));
K(i,j)=(sum((dummy1+dummy2+dummy3+dummy4+dummy5+dummy6+dummy7).*area ))/2;
else
dummy1=(B1(:,i).*B1(:,j)+(B1(:,i).*B2(:,j)+...
      B1(:,j).*B2(:,i)).*(aver_x1));
dummy2=(B1(:,i).*B3(:,j) + B1(:,j).*B3(:,i)).*(aver_x2);
dummy3=(B2(:,i).*B3(:,j) + B2(:,j).*B3(:,i)).*((1/12)*...
       ones(nelem,1)).*(difx11.*difx21+...
        difx12.*difx22 +diffx13.*difx23);
dummy4=(B2(:,i).*B3(:,j) + B2(:,j).*B3(:,i)).*(aver_x1.*aver_x2);
dummy5=(B2(:,i).*B2(:,j)).*( difx11.^2 +difx12.^2 +difx13.^2).*...
     ((1/12)*ones(nelem,1));
dummy6=(B2(:,i).*B2(:,j).*(aver_x1.^2))+...
     (B3(:,i).*B3(:,j).*(aver_x2.^2));
dummy7=(B3(:,i).*B3(:,j)).*( diffx21.^2 +difx22.^2 +difx23.^2).*...
```

```
        ((1/12)*ones(nelem,1));
   K(i,j)=sum((dummy1+dummy2+dummy3+dummy4+dummy5+dummy6+dummy7).*area );
    end
  end
end;
Ko = K +K';
P = Ko;
```

Main Algorithms

Finite Element L-spline (FE L-spline) Algorithm

The specific objective of this method is to find f that minimizes

$$S_\lambda\{f(\mathbf{x})\} = \sum_{i=1}^{n}\{f(\mathbf{x}_i) - y_i\}^2 + \lambda\int_\Omega \{\Delta f\}^2 d\mu.$$

The variational formulation of this problem is derived by Ramsay (1999) [21] and the finite element solution $\mathbf{f}$ is computed by solving the following linear system:

$$\begin{bmatrix} \lambda\mathbf{Q} & \mathbf{I_n} \\ \mathbf{P} & -\mathbf{Q^T} \end{bmatrix}\begin{bmatrix} \mathbf{g} \\ \mathbf{f} \end{bmatrix} = \begin{bmatrix} \mathbf{y} \\ \mathbf{0} \end{bmatrix}$$

where $\mathbf{Q} = \int(\psi_{x_1}\psi_{x_1}^T + \psi_{x_2}\psi_{x_2}^T)$ and $\mathbf{P} = \int(\psi\psi^T)$.

⟨*FELSp.m*⟩≡

```
function FELSpline = FELSp(Y, X1, X2,lambda2)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
% RAMSAY'S APPROACH
% INPUTS :
%               Y       = function values
%               X1      = regressor variable 1
%               X2      = regressor variable 2
%               lambda2 = smoothing parameter value
% OUTPUT : FELSpline  = smooth
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
INI     = TriData(Y, X1, X2);
A       = CcWise(INI);
B       = AreaCoef(A, INI);
Q       = MinusLap(B, INI);
C1C2    = DifOp(A,  B, INI);
P = Pobject(A, C1C2, B, INI);
nnodes = INI{6};
Z = INI{2};
bigmat2 =sparse(2*nnodes);
bigvec2 =sparse(2*nnodes,1);
bigmat2 = [ (lambda2)*Q speye(nnodes,nnodes); P -Q];
bigvec2 = [ Y; sparse(nnodes,1)];
coef2 = (bigmat2)\(bigvec2);
smoothed2 = coef2((nnodes+1):(2*nnodes),1);
FELSpline = smoothed2;
```

Finite Element Thin-Plate Spline (TPSFEM) Algorithm

This method minimizes the following functional:

$$S_\lambda\{\boldsymbol{f}(\mathbf{x}), s(\mathbf{x})\} = \sum_{i=1}^{n}\{s(\mathbf{x}_i) - y_i\}^2 + \lambda|\boldsymbol{f}|^2_{\mathcal{H}^1(\Omega)^2}$$

over all $\boldsymbol{f} = (f_1, f_2) \in \mathcal{H}^1(\Omega)^2$ and $s \in \mathcal{H}^2(\Omega)$ such that

$$(\nabla s, \nabla v)_{\mathcal{L}^2(\Omega)^2} = (\boldsymbol{f}, \nabla v)_{\mathcal{L}^2(\Omega)^2} \quad , \forall\, v \in \mathcal{H}^1(\Omega).$$

Having described all the previous subalgorithms, the computation of the finite element thin-plate spline (TPSFEM) smooth f is equivalent to solving the following system of linear equations:

$$\begin{bmatrix} \mathbf{B} & \mathbf{0} & \mathbf{0} & \mathbf{Q}^T \\ \mathbf{0} & \lambda\mathbf{Q} & \mathbf{0} & -\mathbf{C_1}^T \\ \mathbf{0} & \mathbf{0} & \lambda\mathbf{Q} & -\mathbf{C_2}^T \\ \mathbf{Q} & -\mathbf{C_1} & -\mathbf{C_2} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{c} \\ \mathbf{c_1} \\ \mathbf{c_2} \\ \mathbf{l} \end{bmatrix} = \begin{bmatrix} \mathbf{b} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix}$$

where $\mathbf{l}$ is the Lagrange multiplier.

⟨*TPSFEM.m*⟩≡

```
function FEThinPlate = TPSFEM(Y, X1, X2, lambda)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% ANU GUYS' APPROACH
% INPUTS :
%                 Y       = function values
%                 X1      = regressor variable 1
%                 X2      = regressor variable 2
```

```
%                lambda  = smoothing parameter value
% OUTPUT :  FEThinPlate = smooth
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
INI      = TriData(Y, X1, X2);
A        = CcWise(INI);
B        = AreaCoef(A, INI);
Q        = MinusLap(B, INI);
C1C2     = DifOp(A,  B, INI);
G1 = C1C2{1};
G2 = C1C2{2};
L=Q;
nnodes = INI{6};
smoothed = sparse(nnodes,1);
coeff = sparse(nnodes, 1);
%setting up the linear system or concatenating all the
%submatrices to form 1 big matrix with lambda
bigmat =sparse(4*nnodes);
bigvec =sparse(4*nnodes,1);
bmat1=[speye(nnodes,nnodes) sparse(nnodes,nnodes)...
          sparse(nnodes) (L)'];
bmat2=[sparse(nnodes) lambda*L sparse(nnodes) -G1'];
bmat3=[sparse(nnodes) sparse(nnodes) lambda*L -G2'];
bmat4=[ L (-G1)  (-G2) sparse(nnodes) ];
bigmat = [ bmat1; bmat2; bmat3; bmat4];
```

```
bigvec=[Y;sparse(nnodes,1) ; sparse(nnodes,1) ;...
       sparse(nnodes, 1)];
coeff = (bigmat)\(bigvec);
smoothed = coeff(1:nnodes,1);
FEThinPlate = smoothed;
```

Modified Finite Element L-spline (MFE L-spline) Algorithm

This technique slightly differs from FE L-spline. It differs from the way the smooth is being constructed and the way the minimization functional is expressed. The derivation is based on TPSFEM and the functional is given by :

$$S_\lambda\{f(\mathbf{x})\} = \sum_{i=1}^{n}\{f(\mathbf{x}_i) - y_i\}^2 + \lambda\int_\Omega \{\Delta f\}^2 \, d\mu.$$

The computation of the modified finite element L-spline (MFE L-spline) smooth is very similar to finite element L-spline (FE L-spline). The linear system is expressed as follows

$$\begin{bmatrix} 2*\mathbf{B} & -\lambda\mathbf{Q} & 2*\mathbf{Q} \\ -\lambda\mathbf{Q} & \mathbf{0} & 2*\mathbf{P}^T \\ \mathbf{Q} & \mathbf{P} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{f} \\ \mathbf{g} \\ \mathbf{l} \end{bmatrix} = \begin{bmatrix} 2\mathbf{b} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix}$$

where $\mathbf{l}$ is the Lagrange multiplier.

⟨*MFELSp.m*⟩≡

```
function MFELSpline = MFELSp(Y, X1, X2, lambda3)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% MY APPROACH
% INPUTS :
```

```
%                Y       = function values
%                X1      = regressor variable 1
%                X2      = regressor variable 2
%                lambda3  = smoothing parameter value
% OUTPUT :  MFELSpline  = smooth
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
INI     = TriData(Y, X1, X2);
A       = CcWise(INI);
B       = AreaCoef(A, INI);
Q       = MinusLap(B, INI);
C1C2    = DifOp(A,  B, INI);
P = Pobject(A, C1C2, B, INI);
nnodes = INI{6};
bigmat3 =sparse(3*nnodes);
bigvec3 =sparse(3*nnodes,1);
bigmat3 = [2*speye(nnodes) -(lambda3)*Q  2*Q;...
            -lambda3*Q  sparse(nnodes)  2*(P)';...
             Q  P sparse(nnodes) ];
bigvec3 = [2*Y; sparse(nnodes,1) ; sparse(nnodes, 1)];
coef3 = (bigmat3)\(bigvec3);
smoothed3 = coef3(1:nnodes,1);
MFELSpline = smoothed3;
```

University of Alberta Library
0 1620 1799 2619
B45841